Linux: Utilitários de Redes

Eduardo Maroñas Monks

Linux: Utilitários de Redes

Eduardo Maroñas Monks

ISBN: 978-65-01-59122-3

Este livro tem como objetivo disponibilizar um guia de referência rápido em utilitários de rede para auxiliar administradores de sistemas operacionais Linux nas rotinas de administração.

Para saber mais sobre a área de Redes de Computadores:
YOUTUBE.COM/EMMONKS

2025

Sobre o Autor

Possui graduação em Bacharel em Análise de Sistemas pela Universidade Católica de Pelotas (1998), Mestrado em Computação pela Universidade Federal do Rio Grande do Sul (2006) e Doutorado pela Universidade Federal de Pelotas (2023). Começou a atuar na área de redes em 1995 como estagiário na UCPEL, onde trabalhou por 14 anos. No período de 1999 a 2004 e 2006 a 2007, atuou como docente em cursos de Ciência da Computação e Tecnologia em Análise e Desenvolvimento de Sistemas na UCPEL. Em 2007, passou a exercer a função de docente da UniSenac Pelotas, em que esteve na coordenação da criação do curso superior em Redes de Computadores, curso que atingiu a nota máxima no ENADE de 2017. Em 2014 tornou-se Analista de TI na UFPEL, atuando na administração de redes. Possui certificados Cisco CCAI, Cisco CCNA e LPIC-1.

Para saber mais: Canal do Youtube criado em 2009 com mais de 600 vídeos sobre a área de rede, disponível em **http://www.youtube.com/emmonks**

Aviso Importante

Sobre esta Obra

Este livro pode ser usado de forma livre, desde que os créditos e as referências sejam publicados. As informações contidas no documento são de minha autoria, fruto de experiências de mais de 25 anos atuando na área. Nos casos onde são apontados ferramentas ou sites, poderá haver inconsistências no acesso futuro e poderão causar indisponibilidades.

Se este livro tiver ajudado de alguma forma, por favor, contribua com o valor que achar justo como forma de agradecimento, um cafezinho está ótimo. Obrigado!

Chave do PIX: **7f7fa84a-6ce9-45da-a42f-a77d18eb232c**

Este livro também pode ser obtido em formato físico na loja da editora UICLAP, disponível em **https://loja.uiclap.com/**

Sumário

1 Introdução 9
1.1 Administração de Sistemas Linux 9
1.2 Ambiente para Testes 9
1.3 Escolha da Distribuição 9

2 Comandos Básicos 11
2.1 ifconfig 11
2.2 ip 12
2.3 arp 14
2.4 netstat 14
2.5 route 15
2.6 ethtool 16
2.7 dig 16
2.8 host 17
2.9 nslookup 18
2.10 ping 19
2.11 traceroute 19
2.12 iperf 20
2.13 arping 21
2.14 wget 21
2.15 ssh e scp 22
2.16 ss 22
2.17 mtr 23
2.18 dhclient 24
2.19 telnet 24
2.20 netcat (nc) 24
2.21 curl 25
2.22 rysnc 26
2.23 iwconfig 26
2.24 iwlist 27
2.25 ntpdate 27
2.26 iftop 27
2.27 arp-scan 28

2.28 ftp 28
2.29 whois 29
2.30 mutt 29
2.31 mail / mailx 29
2.32 iperf3 30
2.33 kdig 31
2.34 lldpcli 32
2.35 ldap* 33
2.36 mii-tool 34
2.37 arpwatch 34
2.38 nethogs 35
2.39 tracepath 35
2.40 speedtest-cli 35
2.41 lsof 36
2.42 bmon 36

3 Procedimentos 37
3.1 Renomear interface 37
3.2 Uso de VLANs 37
3.3 Interface em Bridge 38
3.4 Interface em Bonding 40
3.5 Configurações de interface em arquivos 42

4 Utilitários Avançados 45
4.1 tcpdump 45
4.2 socat 47
4.3 sshpass 48
4.4 Python Simple HTTP Server 48
4.5 comandos r (rcp, rexec, rlogin, and rsh) 49
4.6 net 49
4.7 snmp* 50
4.8 nfs* 50
4.9 sshfs 52
4.10 nmap 52
4.11 ngrep 54
4.12 tshark 55
4.13 nmcli 59

4.14 iscsi* 62
4.15 sysctl - parâmetros de rede 65
4.16 hping3 67
4.17 ip tunnel 68

5 Iptables . . . 73
5.1 Histórico 73
5.2 Componentes 73
5.2.1 Chains . . . 73
5.2.2 Tabelas . . . 74
5.2.3 Políticas . . . 74
5.2.4 Ações . . . 74
5.3 Manipulação de Regras 75
5.4 Valores padrão 75
5.5 Módulos 75
5.6 Outros Exemplos 76
5.7 Exemplo de Script de Firewall Completo 76
5.7.1 Firewall da Empresa 1 . . . 76
5.7.2 Firewall da Empresa 2 . . . 79
5.7.3 Firewall da Empresa 1 - versão IPv6 . . . 81

6 Docker Networking . . . 83

7 Anexo . . . 87
7.1 systemctl 87
7.2 journalctl 88
7.3 Vmware Player: Configurações de Interfaces de Rede 89
7.3.1 Vídeos Relacionados . . . 91
7.4 VirtualBox: Configurações de Interfaces de Rede 91
7.4.1 Vídeos Relacionados . . . 92
7.5 Tabelas auxiliares 94
7.6 Características de Rede 95
7.7 Diagnóstico de Problemas em Rede 95

1. Introdução

Este livro foi concebido baseado na experiência prática e da necessidade de um recurso conciso e direto ao ponto. O seu objetivo primordial é ser um guia prático de consulta, um ajudante indispensável que auxiliará nas rotinas diárias de administração de redes em ambientes Linux. Não foi pretensão do autor gerar uma enciclopédia exaustiva de todos os comandos existentes e detalhes teóricos, mas sim um manual focado na aplicação, apresentando os comandos, utilitários e procedimentos essenciais que todo administrador de sistemas Linux deve ter na sua caixa de ferramentas.

1.1 Administração de Sistemas Linux

A administração de servidores Linux é a rotina da maioria dos administradores de redes e sistemas. Desta forma, dominar os comandos e utilitários essenciais do Linux agilizam o diagnóstico e a resolução dos problemas. Portanto, o primeiro momento é ter conhecimento de quais são os comandos e utilitários essenciais. Em um segundo momento, é explorá-los para adaptá-los às necessidades de cada problema. Para aqueles iniciantes no acesso remoto e no uso do interpretador em linhas de comando é importante assistir os vídeos introdutórios disponíveis em **Acesso remoto em VMs Linux**[1] e **Linux: Dificuldades para os Iniciantes**[2].

1.2 Ambiente para Testes

Para realizar testes com as ferramentas é importante o uso com cenários mais completos. Uma das alternativas é o uso do emulador CORE. Neste emulador é possível criar cenários complexos com centenas de hosts Linux e com o uso das ferramentas apresentadas neste livro. Uma máquina virtual com o emulador CORE está disponível (**link para download da VM**[3]).

1.3 Escolha da Distribuição

As principais distribuições são baseadas em Redhat (CentOS, Oracle Linux) ou em Debian (Ubuntu, Mint). As diferenças principais entre as distribuições estão na forma de gerenciamento de pacotes, na organização do sistema de arquivos com caminhos diferentes e na política de uso. Por exemplo, uma distribuição tal como a Debian possui políticas restritas para o uso de pacotes que não sigam o licenciamento GPLv3 e os não disponibiliza por padrão. Entretanto, a escolha da distribuição a ser usada é regida muito mais por uma questão de gosto pessoal ou por necessidade de homologação devido a alguma aplicação a ser instalada ou algum hardware específico. Para os iniciantes na instalação de máquinas virtuais com o sistema operacional Linux podem obter ajuda por meio dos vídeos disponíveis em **CentOS 8: instalação em VM**[4] e **Debian 11: instalação em VM**[5].

[1] https://www.youtube.com/watch?v=JwgP_8jQuLY
[2] https://www.youtube.com/watch?v=cibYsdJAQOE
[3] https://tinyurl.com/ymmmd8fy
[4] https://www.youtube.com/watch?v=_mYl4qOIWWo
[5] https://www.youtube.com/watch?v=iPsIP1HjUlQ

Os utilitários tratados neste livro estão disponíveis em qualquer distribuição atual e os parâmetros dos exemplos deverão funcionar sem problemas.

2. Comandos Básicos

Neste capítulo são apresentados os comandos básicos para configuração de interfaces de rede. Uma das exceções aos comandos básicos é o comando **ip** que engloba diversas funcionalidades para administração de redes em Linux.

2.1 ifconfig

[1]

Descrição 2.1 Comando para realizar configurações na interface de rede e obter estatísticas diversas. Em distribuições atuais vem sendo substituído pelo comando **ip**. Para ativar os utilitários legados de rede em distribuições Debian/Ubuntu, usar o comando "apt install net-tools".

- Para mostrar todas as interfaces disponíveis
  - **Exemplo 2.1** *ifconfig -a*
- Para mostrar informações sobre a interface eth0
  - **Exemplo 2.2** *ifconfig eth0*
- Para configurar um IP na interface eth0
  - **Exemplo 2.3** *ifconfig eth0 10.0.0.2 netmask 255.255.255.0*
  - **Exemplo 2.4** *ifconfig eth0 10.0.0.2/24*
- Trocar o endereço físico da interface eth0
  - **Exemplo 2.5** *ifconfig eth0 hw ether 00:cc:00:ff:ff:ee*
- Para criar um outro IP na interface eth0 (IP alias)
  - **Exemplo 2.6** *ifconfig eth0:1 10.10.0.2/24*
- Para modificar o MTU da interface eth0
  - **Exemplo 2.7** *ifconfig eth0 mtu 9000*
- Para remover um IP da interface eth0
  - **Exemplo 2.8** *ifconfig eth0 0.0.0.0*
- Para desativar a interface eth0
  - **Exemplo 2.9** *ifconfig eth0 down*
- Para desativar um IP alias
  - **Exemplo 2.10** *ifconfig eth0:1 down*
- Para ativar a interface eth0
  - **Exemplo 2.11** *ifconfig eth0 up*

[1] https://www.youtube.com/watch?v=giweVMR4TYE

- Para adicionar um endereço IPv6 na interface eth0
  - **Exemplo 2.12** *ifconfig eth0 inet6 add 2001:0db8:0:200::3/64*
- Para remover um endereço IPv6 na interface eth0
  - **Exemplo 2.13** *ifconfig eth0 inet6 del 2001:0db8:0:200::3/64*

2.2 ip

[2]

Descrição 2.2 Comando para realizar configurações na interface de rede, roteamento, tunelamento, obter estatísticas e outras diversas funcionalidades. Em distribuições atuais o comando **ip** está disponibilizado como principal ferramenta de configuração de interfaces de rede.

- Para listar todas as interfaces
  - **Exemplo 2.14** *ip link show*
- Para ativar a interface eth0
  - **Exemplo 2.15** *ip link set eth0 up*
- Para desativar a interface eth0
  - **Exemplo 2.16** *ip link set eth0 down*
- Para mostrar o endereçamento das interfaces
  - **Exemplo 2.17** *ip addr show*
- Para mostrar o endereçamento da interface eth0
  - **Exemplo 2.18** *ip addr show dev eth0*
- Para mostrar os hosts vizinhos (conectados na mesma rede física que tenham se comunicado com o host e estejam na tabela ARP)
  - **Exemplo 2.19** *ip neigh show*
  - **Exemplo 2.20** *ip neigh show dev eth0*
- Para adicionar uma entrada na tabela ARP
  - **Exemplo 2.21** *ip neigh add 192.168.1.1 lladdr 00:cc:00:ff:ff:ee dev eth0*
- Para remover uma entrada da tabela ARP
  - **Exemplo 2.22** *ip neigh del 192.168.1.1 dev eth0*
- Limpar toda a tabela ARP de uma VLAN
  - **Exemplo 2.23** *ip -s neigh flush dev eth1.212*
- Para adicionar mais um IP na interface eth0 (similar ao IP alias do ifconfig)
  - **Exemplo 2.24** *ip addr add 192.168.1.2/24 dev eth0*
- Para remover endereços adicionais no eth0
  - **Exemplo 2.25** *ip addr del 192.168.1.5/24 dev eth0*

[2] https://www.youtube.com/watch?v=giweVMR4TYE

- Para configurar um endereço IP na interface eth0
  - **Exemplo 2.26** *ip addr add 1.2.3.4/24 broadcast 1.2.3.255 dev eth0*
- Para remover um endereço IP na interface eth0
  - **Exemplo 2.27** *ip addr del 1.2.3.4/24 broadcast 1.2.3.255 dev eth0*
- Para trocar o endereço físico da interface eth0
  - **Exemplo 2.28** *ip link set dev eth0 down*
  - **Exemplo 2.29** *ip link set dev eth0 address 00:cc:00:ff:ff:ee*
  - **Exemplo 2.30** *ip link set dev eth0 up*
- Para modificar o MTU da interface para o valor 1476
  - **Exemplo 2.31** *ip link set mtu 1476 dev eth0*
- Para listar a tabela de roteamento
  - **Exemplo 2.32** *ip route show*
- Para adicionar uma rota padrão IPv4
  - **Exemplo 2.33** *ip route add default via 192.168.1.254*
- Para remover uma rota padrão IPv4
  - **Exemplo 2.34** *ip route del default via 192.168.1.254*
- Para adicionar uma rota estática IPv4
  - **Exemplo 2.35** *ip route add 192.168.1.0/24 dev eth0*
  - **Exemplo 2.36** *ip route add 192.168.8.0/24 via 192.168.254.254*
- Para remover uma rota estática IPv4
  - **Exemplo 2.37** *ip route del 192.168.1.0/24 dev eth0*
  - **Exemplo 2.38** *ip route del 192.168.8.0/24 via 192.168.254.254*
- Para mostrar os endereços IPv6 de todas as interfaces
  - **Exemplo 2.39** *ip -6 a*
- Para adicionar um endereço IPv6 na interface eth0
  - **Exemplo 2.40** *ip -6 addr add 2001:0db8:0:200::3/64 dev eth0*
- Para remover um endereço IPv6 na interface eth0
  - **Exemplo 2.41** *ip -6 addr del 2001:0db8:0:200::3/64 dev eth0*
- Para adicionar uma rota default IPv6
  - **Exemplo 2.42** *ip -6 route add default via 2001:0db8:0:200::1*
- Para remover uma rota default IPv6
  - **Exemplo 2.43** *ip -6 route del default via 2001:0db8:0:200::1*
- Para adicionar uma rota estática IPv6
  - **Exemplo 2.44** *ip -6 route add 2001:0db8:0:201::/64 via 2001:0db8:0:200::1*
- Para remover uma rota estática IPv6
  - **Exemplo 2.45** *ip -6 route del 2001:0db8:0:201::/64 via 2001:0db8:0:200::1*

- Para listar a tabela de roteamento IPv6
  - **Exemplo 2.46** *ip -6 route show*
- Para listar a tabela de vizinhos
  - **Exemplo 2.47** *ip -6 neigh show*
- Para adicionar as marcações de VLANs. No exemplo é adicionada a VLAN ID 100 na interface eth1, criando a interface eth1.100. Para criar múltiplas interfaces marcadas basta repetir o comando variando o name (nome) e o vlan id.
  - **Exemplo 2.48** *ip link add link eth1 name eth1.100 type vlan id 100*
- Para remover uma interface marcada para uso de VLANs. No exemplo a remoção da interface eth1.100
  - **Exemplo 2.49** *ip link delete eth1.100*

2.3 arp

[3]

Descrição 2.3 O protocolo ARP (*Address Resolution Protocol*) tem como principal função a tradução de endereços de forma dinâmica na rede. O uso mais comum é prover o processo da tradução de endereços IP e de endereços físicos (MAC address). O Comando **arp** permite gerenciar as tabelas de tradução de endereços e é bastante útil para descoberta de *hosts* e na administração de serviços de rede.

- Para listar a tabela ARP
  - **Exemplo 2.50** *arp -an*
- Para remover uma entrada na tabela ARP
  - **Exemplo 2.51** *arp -i eth0 -d 192.168.1.1*
- Para adicionar uma entrada na tabela ARP de forma estática
  - **Exemplo 2.52** *arp -s 192.168.1.1 00:cc:00:ff:ff:ee*

2.4 netstat

[4]

Descrição 2.4 Comando para realizar análises e amostragens das conexões de rede, tabelas de roteamento e estatísticas gerais. O comando **ss** é a opção em distribuições mais atuais.

- Para mostrar todas as portas TCP e UDP em escuta e os processos responsáveis
  - **Exemplo 2.53** *netstat -nutlp*
- Para listar a tabela de roteamento

[3] https://www.youtube.com/watch?v=Fiz2iUJ7YAA
[4] https://www.youtube.com/watch?v=OUveXtVdYYY

- **Exemplo 2.54** *netstat -rn*

- Para mostrar todas as conexões TCP abertas
  - **Exemplo 2.55** *netstat -tlnp*
- Para mostrar todas as conexões TCP abertas, de forma contínua
  - **Exemplo 2.56** *netstat -tlnpc*
- Para mostrar todas as conexões TCP abertas, modo estendido
  - **Exemplo 2.57** *netstat -tulpen*
- Para listar todas as conexões
  - **Exemplo 2.58** *netstat -a*
- Para listar estatísticas por protocolo
  - **Exemplo 2.59** *netstat -s*
- Para listar estatísticas somente do protocolo UDP
  - **Exemplo 2.60** *netstat -su*
- Para listar estatísticas somente do protocolo TCP
  - **Exemplo 2.61** *netstat -st*
- Para listar estatísticas das interfaces
  - **Exemplo 2.62** *netstat -i*
- Para obter informações sobre os temporizadores das conexões
  - **Exemplo 2.63** *netstat -o*
- Para listar a tabela de roteamento IPv6
  - **Exemplo 2.64** *netstat -rn -A inet6*

2.5 route

[5]

Descrição 2.5 Comando para mostrar informações e gerenciar tabelas de roteamento em IPv4 e IPv6. Este comando vem sendo substituído pelo **ip** em distribuições mais atuais.

- Para listar a tabela de roteamento
  - **Exemplo 2.65** *route*
- Para adicionar a rota padrão
  - **Exemplo 2.66** *route add default gw 192.168.1.1*
- Para adicionar uma rota estática
  - **Exemplo 2.67** *route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0*
  - **Exemplo 2.68** *route add -net 192.168.1.0 netmask 255.255.255.0 gw 192.168.1.254*
- Para remover rotas estáticas

[5] https://www.youtube.com/watch?v=Ej8ZpdkTtZM

- **Exemplo 2.69** *route del -net 192.168.1.0 netmask 255.255.255.0 gw 192.168.1.254*

- Para remover a rota padrão
  - **Exemplo 2.70** *route del default gw 192.168.1.1*
- Para adicionar uma rota IPv6 padrão
  - **Exemplo 2.71** *route -A inet6 add default gw 2001:0db8:0:200::1*
- Para remover uma rota IPv6 padrão
  - **Exemplo 2.72** *route -A inet6 del default gw 2001:0db8:0:200::1*
- Para listar a tabela de roteamento IPv6
  - **Exemplo 2.73** *route -A inet6*

2.6 ethtool

Descrição 2.6 Comando para gerenciar funções de mais baixo nível da interface de rede, tais como auto-negociação, velocidade, duplex e estatísticas de uso.

- Para mostrar informações gerais sobre a interface eth0
  - **Exemplo 2.74** *ethtool eth0*
- Para mostrar informações sobre o módulo do kernel (driver) da interface eth0
  - **Exemplo 2.75** *ethtool -i eth0*
- Para mostrar estatísticas de tráfego da interface eth0
  - **Exemplo 2.76** *ethtool -S eth0*
- Para mostrar informações sobre configurações de TX, RX e auto-negociação
  - **Exemplo 2.77** *ethtool -a eth0*
- Para fazer piscar o led da placa (ajudar na identificação física da placa)
  - **Exemplo 2.78** *ethtool -p eth0*
- Para forçar 100Mbit/s na interface eth0
  - **Exemplo 2.79** *ethtool -s eth0 speed 100*
- Para desabilitar a auto-negociação na interface eth0
  - **Exemplo 2.80** *ethtool -s eth0 autoneg off*
- Para forçar o modo full-duplex na interface eth0
  - **Exemplo 2.81** *ethtool -s eth0 duplex full*
- Para ativar a opção Wake-on-LAN na interface eth0
  - **Exemplo 2.82** *ethtool -s eth0 wol g*

2.7 dig

[6]

[6] https://www.youtube.com/watch?v=xd622Qaqkyo

Descrição 2.7 Comando para realizar consultas servidores DNS (*Domain Naming System*), alternativa aos comandos **nslookup** e e **host**.

- Para descobrir informações sobre um domínio
  - **Exemplo 2.83** *dig www.senacrs.com.br*
- Para descobrir informações sobre um domínio, com saída reduzida
  - **Exemplo 2.84** *dig www.senacrs.com.br +noall +answer*
- Para descobrir informações sobre um domínio, com saída mínima
  - **Exemplo 2.85** *dig www.senacrs.com.br +short*
- Para descobrir informações sobre os servidores de e-mail de um domínio
  - **Exemplo 2.86** *dig MX senacrs.com.br*
- Para descobrir informações sobre os servidores de DNS de um domínio
  - **Exemplo 2.87** *dig NS senacrs.com.br*
- Para receber informações sobre registros de IPv6 de um domínio
  - **Exemplo 2.88** *dig AAAA senacrs.com.br*
- Para receber o maior número de informações sobre um domínio
  - **Exemplo 2.89** *dig ANY senacrs.com.br*
- Para fazer consulta sobre o DNS reversos para determinado IP
  - **Exemplo 2.90** *dig -x 177.1.214.233*
- Para fazer consulta sobre o DNS reversos para determinado IP, com saída reduzida
  - **Exemplo 2.91** *dig -x 177.1.214.233 +short*
- Para descobrir informações sobre um domínio, consultando outro servidor, no exemplo o servidor 8.8.8.8
  - **Exemplo 2.92** *dig @8.8.8.8 www.senacrs.com.br*
- Para realizar uma consulta completa para determinado domínio, passando pelos servidores raiz
  - **Exemplo 2.93** *dig www.senacrs.com.br +trace*
- Para obter informações sobre o SOA (*Start of Authority*) de um domínio
  - **Exemplo 2.94** *dig SOA senacrs.com.br*
- Para obter informações sobre configurações e validações de SPF de um domínio
  - **Exemplo 2.95** *dig TXT senacrs.com.br*

2.8 host

Descrição 2.8 Comando para realizar consultas servidores DNS (*Domain Naming System*), alternativa aos comandos **nslookup** e **dig**.

- Para descobrir informações sobre um domínio
  - **Exemplo 2.96** *host www.senacrs.com.br*
- Para descobrir informações sobre um domínio, com saída aumentada
  - **Exemplo 2.97** *host -v www.senacrs.com.br*

- Para descobrir informações sobre os servidores de e-mail de um domínio
  - **Exemplo 2.98** *host -t MX senacrs.com.br*
- Para descobrir informações sobre os servidores de DNS de um domínio
  - **Exemplo 2.99** *host -t NS senacrs.com.br*
- Para descobrir informações sobre o IPv6 de um domínio
  - **Exemplo 2.100** *host -t AAAA senacrs.com.br*
- Para receber o maior número de informações sobre um domínio
  - **Exemplo 2.101** *host -t ANY senacrs.com.br*
- Para fazer consulta sobre o DNS reversos para determinado IP
  - **Exemplo 2.102** *host 177.1.214.233*
- Para descobrir informações sobre um domínio, consultando outro servidor, no exemplo o servidor 8.8.8.8
  - **Exemplo 2.103** *host www.senacrs.com.br 8.8.8.8*
- Para obter informações sobre o SOA (*Start of Authority*) de um domínio
  - **Exemplo 2.104** *host -t SOA senacrs.com.br*
- Para obter informações sobre configurações e validações de SPF de um domínio
  - **Exemplo 2.105** *host -t TXT senacrs.com.br*

2.9 nslookup

[7]

Descrição 2.9 Comando para realizar consultas servidores DNS (*Domain Naming System*), alternativa aos comandos **host** e **dig**.

- Para descobrir informações sobre um domínio
  - **Exemplo 2.106** *nslookup www.senacrs.com.br*
- Para descobrir informações sobre os servidores de e-mail de um domínio
  - **Exemplo 2.107** *nslookup -query=MX senacrs.com.br*
- Para descobrir informações sobre os servidores de DNS de um domínio
  - **Exemplo 2.108** *nslookup -query=NS senacrs.com.br*
- Para receber o maior número de informações sobre um domínio
  - **Exemplo 2.109** *nslookup -query=ANY senacrs.com.br*
- Para receber informações sobre IPv6 de um domínio
  - **Exemplo 2.110** *nslookup -query=AAAA senacrs.com.br*
- Para fazer consulta sobre o DNS reversos para determinado IP
  - **Exemplo 2.111** *nslookup 177.1.214.233*

[7] https://www.youtube.com/watch?v=xd622Qaqkyo

- Para descobrir informações sobre um domínio, consultando outro servidor, no exemplo o servidor 8.8.8.8
  - **Exemplo 2.112** *nslookup www.senacrs.com.br 8.8.8.8*
- Para obter informações sobre o SOA (Start of Authority) de um domínio
  - **Exemplo 2.113** *nslookup -query=SOA senacrs.com.br*
- Para obter informações sobre configurações e validações de SPF de um domínio
  - **Exemplo 2.114** *nslookup -query=TXT senacrs.com.br*

2.10 ping

[8]

Descrição 2.10 Comando para realizar testes de conectividade e condições da rede. Funciona baseado no protocolo ICMP e possibilita obter diagnósticos de perdas, atrasos e alcance entre hosts.

- Para realizar teste contínuo
  - **Exemplo 2.115** *ping www.senacrs.com.br*
- Para realizar teste com 20 pacotes
  - **Exemplo 2.116** *ping -c 20 www.senacrs.com.br*
- Para realizar teste com pacotes de 1000 Bytes
  - **Exemplo 2.117** *ping -s 1000 www.senacrs.com.br*
- Para realizar teste com pacotes enviados no intervalo de 0,5 segundos
  - **Exemplo 2.118** *ping -i 0.5 www.senacrs.com.br*
- Para realizar teste com pacotes no modo flood (inundação)
  - **Exemplo 2.119** *ping -f www.senacrs.com.br*
- Para utilizar IPv6
  - **Exemplo 2.120** *ping6 www.senacrs.com.br*

2.11 traceroute

[9]

Descrição 2.11 Comando para realizar testes de conectividade, condições da rede e traçar o caminho (rota) entre uma origem e um destino. O **traceroute** é baseado no protocolo ICMP e no campo TTL (*Time to Live*) do protocolo IP que é decrementado a cada passagem por um roteador e ao chegar a zero retorna uma mensagem ICMP para a origem. Em Linux, por padrão, são usadas portas altas UDP para verificar o caminho do pacote por meio de

[8] https://www.youtube.com/watch?v=Fiz2iUJ7YAA
[9] https://www.youtube.com/watch?v=Fiz2iUJ7YAA

mensagens ICMP *Destination Port Unreachable* (este comportamento pode ser bloqueado por firewalls intermediários).

- Para realizar testes de rota básico para determinado destino
  - **Exemplo 2.121** traceroute www.senacrs.com.br
- Para realizar testes de rota básico para determinado destino, em IPv6
  - **Exemplo 2.122** *traceroute6 www.senacrs.com.br*
- Para realizar testes de rota básico para determinado destino, sem resolução de nomes (DNS)
  - **Exemplo 2.123** *traceroute -n www.senacrs.com.br*
- Para realizar testes de rota básico para determinado destino, com o protocolo ICMP ao invés do UDP padrão
  - **Exemplo 2.124** *traceroute -I www.senacrs.com.br*

2.12 iperf

[10]

Descrição 2.12 Ferramenta para realizar testes de vazão entre dois ou mais hosts. O **iperf** permite a utilização do protocolo TCP e UDP para realizar os testes de vazão que tem como objetivo medir o desempenho da rede.

- Para colocar em modo servidor, com o protocolo TCP e a porta padrão 5001
  - **Exemplo 2.125** *iperf -s*
- Para colocar em modo servidor, com o protocolo TCP e a porta padrão 5001, com IPv6
  - **Exemplo 2.126** *iperf -s -V*
- Para executar o cliente, com o protocolo TCP, 10 segundos de teste e a porta padrão 5001
  - **Exemplo 2.127** *iperf -c IP_Servidor*
- Para executar o cliente, com o protocolo TCP, 10 segundos de teste e a porta padrão 5001, com IPv6
  - **Exemplo 2.128** *iperf –V -c IP_Servidor*
- Para executar o cliente, com o protocolo TCP, 30 segundos de teste, relatórios a cada 1s e a porta padrão 5001
  - **Exemplo 2.129** *iperf -c IP_Servidor -i 1 -t 30*
- Para colocar em modo servidor, com o protocolo TCP e a porta 15001
  - **Exemplo 2.130** *iperf -s -p 15001*
- Para colocar em modo servidor, com o protocolo UDP e a porta padrão 5001

[10] https://www.youtube.com/watch?v=djgi6_mu338

■ **Exemplo 2.131** *iperf -s -u*

- Para executar o cliente, com o protocolo UDP, 30 segundos de teste, relatórios a cada 1s e a porta 15001

  ■ **Exemplo 2.132** *iperf -c IP_Servidor -i 1 -t 30 -u -p 15001*

- Para executar o cliente, com o protocolo TCP, 30 segundos de teste, relatórios a cada 1s, a porta padrão 5001 e com 10 conexões em paralelo

  ■ **Exemplo 2.133** *iperf -c IP_Servidor -i 1 -t 30 -P 10*

2.13 arping

Descrição 2.13 Comando para realizar consultas por meio do protocolo ARP (*Address Resolution Protocol*), similar a ferramenta **ping**. O objetivo do uso é descobrir se um determinado endereço físico está ativo em rede.

- Para enviar requisições ARP para um host vizinho pela interface eth0. Caso o host que tenha o IP 192.168.1.1 esteja ativo na rede, haverá o retorno com a resposta e o tempo de latência.

  ■ **Exemplo 2.134** *arping -I eth0 192.168.1.1*

- Para procurar endereços IP duplicados usando mensagens de ARP Gratuito. Este comando envia mensagens ARP solicitando quem teria o endereço físico do endereço IP 192.168.1.1. O IP 192.168.1.1 seria um endereço IP conhecido e se estaria buscando mais alguma interface com o mesmo IP. Em caso de IPs duplicados seriam informados os endereços físicos dos hosts.

  ■ **Exemplo 2.135** *arping -D -I eth0 192.168.1.1*

2.14 wget

[11]

Descrição 2.14 É um cliente em linha de comando do protocolo HTTP e HTTPS. Permite que sejam feitos acessos a servidores HTTP e downloads de arquivos (uso mais comum).

- Para fazer o download de uma URL

  ■ **Exemplo 2.136** *wget http://192.168.200.3/arquivo.iso*

- Para fazer o download de uma URL que possua usuário e senha

  ■ **Exemplo 2.137** *wget --http-user=aluno --http-password=senha \ http://192.168.200.3/arquivo.iso*

- Para fazer o download de uma URL por meio de um proxy

  ■ **Exemplo 2.138** *wget -e use_proxy=yes -e http_proxy=192.168.200.253:8080 \ http://192.168.200.3/arquivo.iso*

- Exportar as variáveis do shell http_proxy e https_proxy para uso com web proxy.

[11] https://www.youtube.com/watch?v=Ngvs9faN2T8

■ **Exemplo 2.139** *export http_proxy="http://192.168.200.253:8080"*
export https_proxy="http://192.168.200.253:8080"

2.15 ssh e scp

[12]

Descrição 2.15 Comandos com clientes de SSH e SCP para realizar acesso remoto e cópia de arquivos por meio do protocolo SSH. Estas ferramentas são fundamentais para o gerenciamento de servidores Linux.

- Para acessar um servidor SSH (192.168.200.3), na porta padrão, com o usuário aluno.

  ■ **Exemplo 2.140** *ssh aluno@192.168.200.3*

- Para acessar um servidor SSH (192.168.200.3), na porta padrão, com o usuário aluno e obter o modo de depuração.

  ■ **Exemplo 2.141** *ssh -vvvv aluno@192.168.200.3*

- Para acessar um servidor SSH (192.168.200.3), na porta 34000, com o usuário aluno.

  ■ **Exemplo 2.142** *ssh -p34000 aluno@192.168.200.3*

- Para acessar um servidor SSH (192.168.200.3), na porta padrão, com o usuário aluno e executar o comando "dig @8.8.8.8 www.senacrs.com.br".

  ■ **Exemplo 2.143** *ssh aluno@192.168.200.3 "dig @8.8.8.8 www.senacrs.com.br"*

- Para copiar o diretório /opt/arquivos do servidor remoto para o diretório local /var/opt, com a porta padrão e com o usuário aluno, mantendo as permissões dos arquivos e diretórios remotos.

  ■ **Exemplo 2.144** *scp -p -r aluno@192.168.200.3:/opt/arquivos /var/opt*

- Para copiar o diretório local /tmp/relatorio para o servidor remoto no diretório /home/aluno, com a porta 34000 e com o usuário aluno, mantendo as permissões dos arquivos e diretórios locais.

  ■ **Exemplo 2.145** *scp -P34000 -p -r /tmp/relatorio aluno@192.168.200.3:~*

2.16 ss

[13]

Descrição 2.16 Comando para realizar análises e amostragens das conexões de rede, tabelas de roteamento e estatísticas gerais. O comando **netstat** é a opção em distribuições mais antigas.

[12] https://www.youtube.com/watch?v=bh7DwPtYkrY
[13] https://www.youtube.com/watch?v=OUveXtVdYYY

- Para mostrar todas as portas TCP e UDP em escuta e os processos responsáveis.
  - **Exemplo 2.146** *ss -tupl*
- Para mostrar todas as conexões TCP abertas.
  - **Exemplo 2.147** *ss -t -a*
- Para mostrar todas as conexões UDP abertas.
  - **Exemplo 2.148** *ss -u -a*
- Para mostrar todas as conexões TCP abertas, de forma contínua a cada 5s.
  - **Exemplo 2.149** *watch -n 5 "ss -t -a"*
- Para listar todas as conexões.
  - **Exemplo 2.150** *watch -n 5 "ss -t -a"*
- Para listar todas as conexões.
  - **Exemplo 2.151** *ss -an*
- Para listar estatísticas por protocolo.
  - **Exemplo 2.152** *ss -sa*
- Para obter informações sobre os temporizadores das conexões.
  - **Exemplo 2.153** *ss -o*

2.17 mtr

[14]

Descrição 2.17 Comando para realizar consultas a roteadores e traçar a rota entre dois hosts. O **mtr** é uma versão do utilitário **traceroute** com mais opções.

- Para realizar testes de forma contínua para determinado endereço.
  - **Exemplo 2.154** *mtr 8.8.8.8*
- Para realizar testes de forma contínua para determinado endereço, com IPv6.
  - **Exemplo 2.155** *mtr -6 www.google.com*
- Para realizar testes por 10 vezes e gerar um relatório.
  - **Exemplo 2.156** *mtr -r -c 10 8.8.8.8*
- Para realizar testes de forma contínua para determinado endereço, sem resolução de nomes (DNS).
  - **Exemplo 2.157** *mtr -n 8.8.8.8*

[14] https://www.youtube.com/watch?v=18KyWTzZfd0

2.18 dhclient

[15]

Descrição 2.18 Comando que implementa um cliente do protocolo DHCP e permite gerenciar o empréstimo de endereços IP.

- Para renovar (*renew*) o IP por DHCP na interface eth0.
  - **Exemplo 2.158** *dhclient eth0*
- Para liberar (*release*) o IP por DHCP na interface eth0.
  - **Exemplo 2.159** *dhclient -r eth0*

2.19 telnet

[16]

Descrição 2.19 Comando que implementa um cliente do protocolo Telnet. É usado para realizar testes em portas de serviços que utilizam o protocolo TCP.

- Para fazer uma conexão à porta 80 de um endereço IP.
  - **Exemplo 2.160** *telnet 19.168.200.3 80*
- Obs.: para cancelar a conexão, utilizar a combinação de teclas "CTRL+Ç" ('^]')

2.20 netcat (nc)

[17]

Descrição 2.20 Comando que provê diversas funcionalidades relacionadas as conexões de rede e para diagnóstico de serviços.

- Para criar um servidor, com o protocolo TCP, na porta 8000.
  - **Exemplo 2.161** *nc -l 8000*
- Para conectar em um servidor, com o protocolo TCP, na porta 8000.
  - **Exemplo 2.162** *nc 192.168.200.3 8000*
- Para criar um servidor, com o protocolo UDP, na porta 8000.
  - **Exemplo 2.163** *nc -u -l 8000*
- Para conectar em um servidor, com o protocolo UDP, na porta 8000.
  - **Exemplo 2.164** *nc -u 192.168.200.3 8000*

[15] https://www.youtube.com/watch?v=lzwtJzw4SXM
[16] https://www.youtube.com/watch?v=2EQzJpI2cag
[17] https://www.youtube.com/watch?v=2EQzJpI2cag

- Para criar um servidor, com o protocolo TCP, na porta 8000, e manter o socket aberto depois da primeira conexão.
  - **Exemplo 2.165** *nc -k -l 8000*
- Para transmitir um arquivo do lado do cliente para o lado do servidor.
  - **Exemplo 2.166** *Servidor: nc -l 8000 > /tmp/arquivo.dat*
  - **Exemplo 2.167** *Cliente: nc 192.168.200.3 8000 < arquivo.dat*
- Para transmitir um arquivo do lado do servidor para o lado do cliente.
  - **Exemplo 2.168** *Servidor: nc -l 8000 < /tmp/arquivo.dat*
  - **Exemplo 2.169** *Cliente: nc 192.168.200.3 8000 > arquivo.dat*
- Para copiar um diretório **com** compactação, diretório corrente no cliente para o diretório corrente no servidor com o IP 10.20.12.254 na porta 4444
  - **Exemplo 2.170** *Servidor: nc -l -p 4444| tar xzv*
  - **Exemplo 2.171** *Cliente: tar czv . | nc -q 0 10.20.12.254 4444*
- Para copiar um diretório **sem** compactação, diretório corrente no cliente para o diretório corrente no servidor com o IP 10.20.12.254 na porta 4444
  - **Exemplo 2.172** *Servidor: nc -l -p 4444| tar xv*
  - **Exemplo 2.173** *Cliente: tar cv . | nc -q 0 10.20.12.254 4444*

2.21 curl

Descrição 2.21 É um cliente em linha de comando do protocolo HTTP e HTTPS. Permite que sejam feitos acessos a servidores HTTP e downloads de arquivos (uso mais comum).

- Realizar o download do arquivo.zip.
  - **Exemplo 2.174** *curl https://192.168.254.95/arquivo.zip*
- Realizar o download do arquivo.zip e salvar com o nome de arquivo.zip_bk.
  - **Exemplo 2.175** *curl -o arquivo.zip_bk https://192.168.254.95/arquivo.zip*
- Realizar o download do arquivo.zip, com o limite de 1 Mbit/s.
  - **Exemplo 2.176** *curl --limit-rate 1m -O https://192.168.254.95/arquivo.zip*
- Realizar o download do arquivo "arquivo.zip" utilizando protocolo FTP no servidor 192.168.254.90, por meio do dados de autenticação aluno e senha teste.
  - **Exemplo 2.177** *curl -u aluno:teste ftp://192.168.254.90/arquivo.zip*
- Realizar o download do arquivo "arquivo.zip" na URL http://www.meusite.com.br utilizando um proxy de endereço 192.168.254.1 na porta 3128.
  - **Exemplo 2.178** *curl -x 192.168.254.1:3128 http://www.meusite.com.br/arquivo.zip*
- Realizar o download do arquivo "arquivo.zip" utilizando uma conexão HTTPS e ignorando os erros de certificados SSL.
  - **Exemplo 2.179** *curl --insecure https://www.meusite.com.br/arquivo.zip*

2.22 rysnc

[18]

Descrição 2.22 Utilitário e protocolo para realizar a sincronização de arquivos e diretórios de forma local ou remota.

- Sincroniza arquivos locais em foramto PDF localizados em "/dados" para o host 192.168.254.29 em "/backup" utilizando o usuário root.

  ■ **Exemplo 2.180** *rsync -avz --include '*.pdf' /dados/ root@192.168.254.29:/backup*

- Sincroniza o diretório atual, mostrando o progresso da cópia e possibilitando o resumo da cópia em caso de falhas, limitando em 600 Kbytes/s para o host remoto 192.168.254.95 em /root/teste.

  ■ **Exemplo 2.181** *rsync -avh --partial --progress --bwlimit=600 . \ root@192.168.254.95:/root/teste*

- Sincroniza o diretório local, /dados/arquivos, com o diretório remoto /dados no host 192.168.254.95. O uso do rsync será por meio do protocolo SSH, com o usuário "aluno" no host remoto.

  ■ **Exemplo 2.182** *rsync -avz -e "ssh -o StrictHostKeyChecking=no \ -o UserKnownHostsFile=/dev/null" --progress /dados/arquivos/ \ aluno@192.168.254.95:/dados/*

- Obs.: o diretório arquivos será sincronizado como um subdiretório em /dados

  ■ **Exemplo 2.183** *rsync -avz -e "ssh -o StrictHostKeyChecking=no \ -o UserKnownHostsFile=/dev/null" --progress /dados/arquivos/* \ aluno@192.168.254.95:/dados/*

- Obs.: o conteúdo do diretório arquivos será sincronizado no diretório em /dados

2.23 iwconfig

Descrição 2.23 Comando para realizar configurações na interface de rede sem fios (wireless) e obter estatísticas diversas.

- Para listar todas as interfaces sem fios disponíveis.

  ■ **Exemplo 2.184** *iwconfig -a*

- Para ativar a interface ath0.

  ■ **Exemplo 2.185** *iwconfig ath0 up*

- Para desativar a interface ath0.

  ■ **Exemplo 2.186** *iwconfig ath0 down*

- Para trocar para o canal 11 na interface ath0.

  ■ **Exemplo 2.187** *iwconfig ath0 interface channel 11*

- Para ativar o modo monitor na interface ath0.

  ■ **Exemplo 2.188** *iwconfig ath0 mode monitor*

[18] https://www.youtube.com/watch?v=wqpH79EDn7M

- Para ativar o modo managed (padrão) na interface ath0.
  - **Exemplo 2.189** *iwconfig ath0 mode managed*

2.24 iwlist

Descrição 2.24 Comando que provê funcionalidades de gerenciamento relacionadas a conexões de rede sem fios.

- Para listar os clientes associados ao access point.
  - **Exemplo 2.190** *iwlist peers*
- Para realizar varredura de canais (*site survey*) utilizando a interface ath0.
  - **Exemplo 2.191** *iwlist ath0 scan*
- Para listar os canais disponíveis.
  - **Exemplo 2.192** *iwlist channel*

2.25 ntpdate

[19]

Descrição 2.25 Cliente do protocolo NTP (Network Time Protocol) para ajuste de dia e horário. Para o ajuste ser correto a zona de tempo (timezone) do host deverá estar correta. No horário de Brasília a zona de tempo é a GMT-3.

- Para ajustar o horário do sistema conforme o servidor de NTP 200.132.0.132.
  - **Exemplo 2.193** *ntpdate 200.132.0.132*
- Para apenas solicitar informações sobre dia e horário ao servidor NTP 200.132.0.132.
  - **Exemplo 2.194** *ntpdate –q 200.132.0.132*

2.26 iftop

[20]

Descrição 2.26 Comando para realizar o monitoramento do tráfego em interfaces de rede. Esta ferramenta permite o uso de filtros compatíveis com a biblioteca Libpcap (tcpdump).

- Para monitorar o tráfego de rede na interface eth0.
  - **Exemplo 2.195** *iftop –i eth0*
- Para monitorar o tráfego de rede na interface eth0 que seja relacionado ao endereço IP 192.168.254.254.
  - **Exemplo 2.196** *iftop –i eth0 –f "host 192.168.254.254"*

[19] https://www.youtube.com/watch?v=l-Sf92uJdZc
[20] https://www.youtube.com/watch?v=0sOAVZcQU9E

- Para monitorar o tráfego de rede na interface eth2 que seja relacionado a rede 192.168.254.0/24 e na porta TCP 8080.

  ■ **Exemplo 2.197** *iftop –i eth0 –f "net 192.168.254 and tcp port 8080"*

2.27 arp-scan

[21]

Descrição 2.27 Comando para realizar descobertas de IPs e hosts ativos na rede forçando mensagens ARP.

- Realiza uma varredura na rede da interface eth1, 10.10.16.0/24 e filtra pelo IP 10.10.16.71

  ■ **Exemplo 2.198** *arp-scan -I eth1 10.10.16.0/24 | grep 10.10.16.71*

- Realiza uma varredura na rede da primeira interface do sistema, buscando hosts ativos que respondam as requisições ARP (*Address Resolution Protocol*

  ■ **Exemplo 2.199** *arp-scan -l*

2.28 ftp

[22]

Descrição 2.28 Cliente do protocolo FTP (File Transfer Protocol) utilizado para a transferência de arquivos. O cliente de FTP possui um interpretador de comandos que possibilita gerenciar as transferências e a organização de arquivos e pastas locais e remotas durante a conexão.

- Acesso ao servidor de FTP ftp.scene.org . Em caso de acesso a um servidor de FTP público é padrão utilizar o usuário anonymous (anônimo) e a senha o e-mail do usuário. Para acessar o diretório público (/pub) e listar o conteúdo do mesmo (ls) e encerra a conexão (quit)

  ■ **Exemplo 2.200** *ftp ftp.scene.org*
  ftp> cd /pub
  ftp> ls
  ftp> quit

- Para realizar o download de um arquivo binário (bin) de nome poweramiga1.zip (get) mostrando a progressão do download (hash)

  ■ **Exemplo 2.201** *ftp>bin*
  ftp>hash
  ftp> get poweramiga1.zip

- Para realizar o upload de todos os arquivos com a extensão .zip (mput *.zip) do diretório local corrente para o diretório remoto corrente no servidor de FTP

[21] https://www.youtube.com/watch?v=jXcbOZPcPDM
[22] https://www.youtube.com/watch?v=gImr3z17eCk

■ **Exemplo 2.202** *ftp>bin*
*ftp> mput *.zip*

2.29 whois

[23]

Descrição 2.29 Comando para realizar pesquisas sobre registros das propriedades de domínios e endereços IP. O resultado do comando mostra informações que podem ser úteis na identificação da origem de conexões.

- Realiza busca de informações sobre o domínio youtube.com

  ■ **Exemplo 2.203** *whois youtube.com*

- Realiza busca de informações sobre o endereço IP 200.18.78.15

  ■ **Exemplo 2.204** *whois 200.18.78.15*

2.30 mutt

Descrição 2.30 Cliente de e-mail em linha de comando para utilizar de forma interativa ou em scripts.

- Envia um e-mail para admin@empresa.br com o assunto "Backup Iniciado" e no corpo da mensagem o texto "Iniciando o backup".

  ■ **Exemplo 2.205** *echo "Iniciando o backup" |mutt -s "Backup Iniciado" \ admin@empresa.br*

- Envia mensagem de monitor@empresa.com.br, nome do emissor "Monitor", para admin@empresa.com.br com o assunto "Torrent", no corpo da mensagem "Uso de Torrent" e o arquivo captura.zip em anexo.

  ■ **Exemplo 2.206**
  ```
  echo "Uso de Torrent"|mutt -a captura.zip \
  -e 'set from=monitor@empresa.com.br realname="Monitor"' \
  -s "Torrent" -- admin@empresa.com.br
  ```

- Acessa a caixa de entrada do usuário johnfoo localizada em /var/spool/mail/johnfoo em linha de comando.

  ■ **Exemplo 2.207** *mutt -f /var/spool/mail/johnfoo*

2.31 mail / mailx

[24]

Descrição 2.31 Cliente de e-mail em linha de comando para utilizar de forma interativa ou em scripts. Para instalar o utilitário no CentOS usar "yum install mailx" e no Ubuntu/Debian/LinuxMint usar "apt install mailutils" . Os comandos mail e mailx apontam para o

[23] https://www.youtube.com/watch?v=4vujUpyGEeE
[24] https://www.youtube.com/watch?v=5PL-HGfaPKc

mesmo utilitário e são aliases.

- Envia mensagem para o usuário johnfoo@mail.com utilizando o envio pelo servidor de SMTPS "smtps.empresa.com.br" na porta 587, com autenticação (usuário no-reply e senha aq123456@). O assunto do e-mail é "Host Infectado".

  ■ **Exemplo 2.208**
  ```
  echo -e "Host Infectado com Malware" | mailx -v \
      -r "no-reply@empresa.com.br" \
      -s "Aviso: Host Infectado " \
      -S smtp="smtps.empresa.com.br:587" \
      -S smtp-use-starttls \
      -S smtp-auth=login \
      -S smtp-auth-user="no-reply" \
      -S smtp-auth-password="aq123456@" \
      -S ssl-verify=ignore \
      johnfoo@mail.com
  ```

- Envia e-mail com o assunto "Manual" para johnfoo@mail.com com o conteúdo do arquivo "corpo-mail.txt" como texto da mensagem.

  ■ **Exemplo 2.209** *mail -s "Manual" johnfoo@mail.com < corpo-mail.txt*

- Envia e-mail com a mensagem redirecionada com o assunto "Aviso"" para os receptores johnfoo@mail.com e para alice@mail.com. O redirecionador com três caracteres "«<" faz com que a mensagem seja enviada direto, sem interação com o usuário.

  ■ **Exemplo 2.210**
  ```
  mail -s "Aviso!" johnfoo@mail.com,alice@mail.com <<< \
  "Por favor, não esquecer de desligar a cafeteira."
  ```

2.32 iperf3

25

Descrição 2.32 O Iperf3 é uma ferramenta para testes de desempenho de redes e é utilizada para medir a largura de banda e a qualidade da conexão de rede. Gera tráfego de rede entre dois pontos para avaliar a velocidade de transferência (vazão). A ferramenta suporta testes tanto em TCP quanto em UDP. Oferece opções avançadas para ajustar configurações de testes. Em comparação com a ferramenta Iperf versão 2, o Iperf3 é uma evolução que possui recursos mais avançados para análise de desempenho de rede. Recomendado para uso em produção e para validação de links com operadoras, utiliza a porta padrão 5201 (a versão do Iperf 2 utiliza a porta 5001 como padrão). As versões 2 e 3/3.1 não possuem compatibilidade.

- Ativar o modo servidor.

  ■ **Exemplo 2.211** *iperf3 -s*

- Ativar o modo cliente (o fluxo de dados padrão é do lado cliente para o lado servidor).

  ■ **Exemplo 2.212** *iperf3 –c IP-Servidor*

- Testes para rodar no servidor, duas instâncias para testes, mostrando resultados

[25] https://www.youtube.com/watch?v=rl6fv3LOaGo

parciais a cada 5 segundos (parâmetro "-i 5"), de download e upload em TCP/UDP. O servidor aceita conexões em TCP, UDP, IPv4 e IPv6 por padrão

- **Exemplo 2.213** *iperf3 -s -i 5 -p 5201*
- **Exemplo 2.214** *iperf3 -s -i 5 -p 5202*

- Para conectar no servidor de endereço IPv4 192.168.0.1, utilizando o protocolo TCP, com largura de banda no limite de 1 Gbit/s (parâmetro "-b 1g"), por 10 segundos (parâmetro "-t 10"), mostrando relatórios parciais a cada 1 segundo (parâmetro "-i 1"). A segunda conexão utiliza o parâmetro -R (reverse) que faz com que o fluxo de dados seja do servidor para o cliente.
  - **Exemplo 2.215** *iperf3 -c 192.168.0.1 -b 1g -t 10 -i 1 -p 5201*
  - **Exemplo 2.216** *iperf3 -c 192.168.0.1 -b 1g -t 10 -i 1 -p 5202 –R*
- Para conectar no servidor de endereço IPv4 192.168.0.1, utilizando o protocolo UDP (parâmetro "-u"), com largura de banda no limite de 1 Gbit/s (parâmetro "-b 1g"), por 10 segundos (parâmetro "-t 10"), mostrando relatórios parciais a cada 1 segundo (parâmetro "-i 1"). A segunda conexão utiliza o parâmetro -R (reverse) que faz com que o fluxo de dados seja do servidor para o cliente.
  - **Exemplo 2.217** *iperf3 -u -c 192.168.0.1 -b 1g -t 10 -i 1 -p 5201*
  - **Exemplo 2.218** *iperf3 -u -c 192.168.0.1 -b 1g -t 10 -i 1 -p 5202 –R*
- Para conectar no servidor de endereço IPv6 2001:0db8:161::100, utilizando o protocolo TCP, com largura de banda no limite de 1 Gbit/s (parâmetro "-b 1g"), por 10 segundos (parâmetro "-t 10"), mostrando relatórios parciais a cada 1 segundo (parâmetro "-i 1"). A segunda conexão utiliza o parâmetro -R (reverse) que faz com que o fluxo de dados seja do servidor para o cliente.
  - **Exemplo 2.219** *iperf3 -6 -c 2001:0db8:161::100 -b 1g -t 10 -i 1 -p 5201*
  - **Exemplo 2.220** *iperf3 -6 -c 2001:0db8:161::100 -b 1g -t 10 -i 1 -p 5202 –R*
- Para conectar no servidor de endereço IPv4 192.168.0.1, na porta 5201, utilizando o protocolo TCP, com largura de banda no limite de 200Mbit/s (parâmetro "-b 200"), por 900 segundos (parâmetro "-t 900"), mostrando relatórios parciais a cada 5 segundos (parâmetro "-i 5") com o uso de 5 threads (parâmetro "-P 5"). O uso de múltiplas threads é um recurso importante para poder validar links simulando múltiplas conexões.
  - **Exemplo 2.221** *iperf3 -c 192.168.0.1 -b 200m -t 900 -i 5 -p 5201 –P 5*
- Para coletar os resultados dos testes, pode-se redirecionar para um arquivo a saída do comando. Nos exemplos, serão criados 4 arquivos, dois para TCP e dois para UDP, em ambos sentidos da conexão (download/upload).
  - **Exemplo 2.222** *iperf3 -c <IP-servidor> -b 1g -t 900 -i 5 -p 5201 > TCP-cliente.txt*
  - **Exemplo 2.223** *iperf3 -c <IP-servidor> -b 1g -t 900 -i 5 -p 5202 -R > TCP-servidor.txt*
  - **Exemplo 2.224** *iperf3 -u -c <IP-servidor> -b 1g -t 900 -i 5 -p 5201 > UDP-cliente.txt*
  - **Exemplo 2.225** *iperf3 -u -c <IP-servidor> -b 1g -t 900 -i 5 -p 5202 \ -R > UDP-servidor.txt*

2.33 kdig

Descrição 2.33 O utilitário kdig é um cliente de DNS, similar ao utilitário **dig**, que faz parte do pacote do servidor de DNS Knot. Para instalar o kdig em distribuições Redhat/CentOS utilizar o comando *yum install knot-utils* e em distribuições Debian/Ubuntu *apt install knotdns-utils*. O kdig é interessante para realizar testes de resolução de DNS com o protocolo DoH (DNS over HTTPS).

- Busca a resolução do domínio youtube.com com consulta ao servidor de DNS 1.1.1.1 utilizando o protocolo DoH (DNS over HTTPS).
  - **Exemplo 2.226** *kdig @1.1.1.1 +https youtube.com*
- Busca a resolução do domínio youtube.com com consulta ao servidor de DNS 1.1.1.1 utilizando o protocolo TCP.
  - **Exemplo 2.227** *kdig @1.1.1.1 +tcp youtube.com*
- Busca a resolução do domínio youtube.com com consulta ao servidor de DNS 1.1.1.1 utilizando o protocolo UDP.
  - **Exemplo 2.228** *kdig @1.1.1.1 youtube.com*
- Busca a resolução do endereço do servidor de e-mail (MX) do domínio globo.com com consulta ao servidor de DNS 8.8.8.8 utilizando o protocolo DoH (DNS over HTTPS).
  - **Exemplo 2.229** *kdig @8.8.8.8 +https globo.com MX*
- Busca a resolução do endereço IPv6 (AAAA) do endereço www.terra.com.br com consulta ao servidor de DNS 1.1.1.1 utilizando o protocolo DoH (DNS over HTTPS).
  - **Exemplo 2.230** *kdig @1.1.1.1 +https www.terra.com.br AAAA*

2.34 lldpcli

[26]

Descrição 2.34 A ferramenta lldpcli faz parte do pacote **lldpd** (https://lldpd.github.io/) [a] que disponibiliza um daemon e um cliente para realizar requisições com o protocolo LLDP (Link Layer Discovery Protocol). O protocolo de descoberta LLDP trabalha na camada de enlace de dados (Layer 2) e possibilita que dispositivos tais como switches, roteadores, impressoras, telefones IP e outros possam informar sobre as capacidades, identificação e, principalmente, informar a respeito de dispositivos vizinhos na rede local. Os arquivos de configuração do daemon ficam em /etc/default/lldpd (Debian/Ubuntu) e /etc/sysconfig/lldpd (CentOS). O cliente **lldpcli** ou **lldpctl** provê as interações com o protocolo em linha de comando.

[a] https://lldpd.github.io/

- Realiza a descoberta de vizinhos
  - **Exemplo 2.231** *lldpcli show neighbors*
- Realiza a descoberta de vizinhos e apresenta os resultados de forma resumida
  - **Exemplo 2.232** *lldpcli show neighbors summary*

[26] https://www.youtube.com/watch?v=B47owGIwBE0

- Realiza a troca do nome do host para "Impressora1" que será divulgado pelo daemon lldpd

  ■ **Exemplo 2.233** *configure system hostname Impressora1*

2.35 ldap*

Descrição 2.35 O LDAP (Lightweight Directory Access Protocol) é um protocolo que permite o acesso a informações de vários sistemas e aplicações em um serviço de diretório. O LDAP é utilizado para gerenciar contas de usuários, configurar a autenticação de serviços de diretório, centralizar informações, reduzir o armazenamento de informações duplicadas e permitir que os usuários tenham um único logon e senha. No Linux, a implementação do protocolo é realizada pelo pacote OpenLDAP (https://www.openldap.org/) que provê diversos utilitários para administração do serviço de diretório.

- Faz a listagem de toda a base LDAP no servidor em localhost e solicita a senha do usuário admin

  ■ **Exemplo 2.234**

```
ldapsearch -D "cn=admin,dc=empresa,dc=com,dc=br" -W -p 389 -h 127.0.0.1 \
-b "dc=empresa,dc=com,dc=br" -s sub -x "(objectclass=*)"
```

- Faz a remoção de toda a unidade organizacional (OU) ou=people,dc=empresa,dc=com,dc=br de forma recursiva, conectando no servidor de IP 192.168.200.3, na porta 1389, passando a senha "senha@123" em linha de comando

  ■ **Exemplo 2.235**

```
ldapdelete -h 192.168.200.3 -p 1389 -c -x -D \
"cn=admin,dc=empresa,dc=com,dc=br" -v  \
-r "ou=people,dc=empresa,dc=com,dc=br" -w senha@123
```

- Faz a remoção do usuário com o uid "aluno10" da unidade organizacional ou=people,dc=empresa,dc=com,dc=br no servidor localhost

  ■ **Exemplo 2.236**

```
 ldapdelete -c -x -D "cn=admin,dc=empresa,dc=com,dc=br" -v
 "uid=aluno10,ou=people,dc=empresa,dc=com,dc=br" -w senha@123
```

- Importa um novo usuário do arquivo dados.ldif no formato LDIF , no servidor em localahost e na porta 1389

  ■ **Exemplo 2.237**

```
ldapadd -h 127.0.0.1 -p 1389 -x -D "cn=admin,dc=empresa,dc=com,dc=br" \
-c -w senha@123 -f dados.ldif

# Exemplo de formato de arquivo LDIF

dn: ou=people,dc=empresa,dc=com,dc=br
objectClass: organizationalUnit
objectClass: top
ou: people

dn: uid=88479439089,ou=people,dc=empresa,dc=com,dc=br
objectClass: person
objectClass: inetOrgPerson
objectClass: brPerson
```

```
objectClass: schacPersonalCharacteristics
cn: FULANO
sn: BARCELOS COSTA DA SILVA
brcpf: 88479439089
schacDateOfBirth: 19651129
mail: fu.lano@gmail.com
userPassword: {SHA}z111jyLHT6oB6GMMgR4tfsvO/wo=
uid: 88479439089
homePostalAddress: ALUNO
```

2.36 mii-tool

Descrição 2.36 O mii-tool é uma ferramenta que exibe e controla o status da interface MII (Media Independent Interface), que é o padrão para comunicação entre um controlador Ethernet e o seu transceptor físico (PHY). Ele é usado principalmente para verificar a velocidade da porta Ethernet, o modo duplex e o status do link. Embora seja uma ferramenta mais antiga, ainda pode ser útil para interfaces mais antigas ou para verificações rápidas. A ferramenta mais moderna que realiza essas funcionalidades é a ethtool.

- Exibe o status atual do link da interface "eth0", incluindo a velocidade (por exemplo, "100baseTx-FD" para 100 Mbit/s Full Duplex) e se o link está ativo.

  ■ **Exemplo 2.238** *mii-tool eth0*

- Força a interface "eth0" a operar em 10 Mbit/s e modo Half Duplex. É útil para testes de compatibilidade ou para isolar problemas de negociação automática.

  ■ **Exemplo 2.239** *mii-tool -F 10baseT-HD eth0*

- Configura a interface "eth0" para retornar ao modo de negociação automática de velocidade e duplex, que é o comportamento padrão e recomendado na maioria dos casos.

  ■ **Exemplo 2.240** *mii-tool -A eth0*

2.37 arpwatch

Descrição 2.37 O arpwatch é um utilitário que monitora o tráfego ARP (Address Resolution Protocol) na rede local e registra alterações no mapeamento de endereços IP para endereços MAC. Ele é uma ferramenta útil para detecção de anomalias, como ataques de ARP spoofing, ou para simplesmente manter um histórico das máquinas que se conectam à rede.

- Ativa em segundo plano (background) o arpwatch para monitorar a interface "eth0". Ele registrará no log do sistema (geralmente em "/var/log/syslog" ou "/var/log/auth.log" dependendo da configuração) quaisquer novas associações IP e MAC ou alterações nas existentes.

  ■ **Exemplo 2.241** *arpwatch -i eth0*

- Além de registrar as mudanças nos logs, esta linha de comando configura o arpwatch para enviar um e-mail para "admin@empresa.com.br" sempre que detectar uma nova entrada ARP ou uma alteração em uma entrada existente. Usar com cuidado esta linha de comando porque podem ser geradas centenas ou até milhares

de mensagens de e-mail dependendo do número de alterações de endereços MAC na rede.

■ **Exemplo 2.242** *arpwatch -i eth0 -m admin@empresa.com.br*

- O parâmetro "-d" (debug) faz com que o arpwatch seja executado em primeiro plano e imprima todas as informações de depuração no terminal, em vez de ir para o log do sistema. Isso é útil para testar a configuração ou solucionar problemas.

  ■ **Exemplo 2.243** *arpwatch -i eth0 -d*

2.38 nethogs

Descrição 2.38 O nethogs é uma ferramenta simples que mostra o consumo de largura de banda por processo em tempo real. Diferente de outras ferramentas que mostram o tráfego por protocolo ou interface, o nethogs identifica qual aplicação ou processo está gerando tráfego de rede.

- Este comando inicia o nethogs e monitora todas as interfaces de rede ativas. Ele exibirá uma lista de processos, o usuário que os executa, a interface de rede e o tráfego de download/upload em KB/s.

  ■ **Exemplo 2.244** *nethogs*

- Ativa as análises de consumo de tráfego na interface eth0. O nethogs filtrará e mostrará apenas o tráfego gerado ou recebido por essa interface.

  ■ **Exemplo 2.245** *nethogs eth0*

2.39 tracepath

Descrição 2.39 O tracepath é uma ferramenta de rastreamento de rota similar ao traceroute, mas com a vantagem de não exigir privilégios de root. Ele tenta descobrir a rota (caminho) que os pacotes IP percorrem de um host local até um host de destino, além de estimar o MTU (Maximum Transmission Unit) do caminho.

- Este comando rastreia a rota dos pacotes do host de origem até o destino "www.google.com". Ele exibirá cada salto (hop) no caminho, o tempo de ida e volta (RTT) e o MTU descoberto para aquele trecho.

  ■ **Exemplo 2.246** *tracepath www.google.com*

- O parâmetro "-p" permite especificar o tamanho do pacote que o tracepath usará para descobrir o MTU do caminho. Isso pode ser útil para depurar problemas de fragmentação de pacotes ou MTU Path Discovery. No exemplo, o rastreamento da rota até o "www.google.com" será executado com pacotes de 1400 bytes, e o MTU final descoberto será refletido na saída.

  ■ **Exemplo 2.247** *tracepath -p 1400 www.google.com*

2.40 speedtest-cli

Descrição 2.40 O speedtest-cli é um cliente de linha de comando não oficial para o popular serviço Speedtest.net, permitindo testar a velocidade da sua conexão de Internet (download, upload e ping) diretamente do terminal Linux.
A ferramenta está disponível em https://www.speedtest.net/apps/cli

- Este comando executa um teste completo, localizando o servidor mais próximo, medindo o ping, a velocidade de download e a velocidade de upload.

  ■ **Exemplo 2.248** *speedtest-cli*

- O primeiro comando com o parâmetro "--list" mostra todos os servidores disponíveis e com o uso do utilitário "grep" foi filtrado pela cidade de "Porto Alegre". O segundo comando "--server ID_DO_SERVIDOR" permite que seja especificado um um servidor de teste usando o ID obtido na lista, o que é útil para testar a conectividade com um local específico.

  ■ **Exemplo 2.249** *speedtest-cli --list | grep "Porto Alegre"* \
  speedtest-cli --server 12345

2.41 lsof

Descrição 2.41 O lsof (List Open Files) é uma ferramenta poderosa para listar todos os arquivos abertos por processos em um sistema Linux. No contexto de redes, ele é indispensável para identificar quais processos estão utilizando portas de rede, quais conexões estão ativas e em qual estado.

- O parâmetro "-i" solicita ao lsof para listar apenas arquivos de rede. Isso inclui sockets TCP e UDP, permitindo ver quais processos estão escutando em portas ou têm conexões estabelecidas. Será gerada uma lista detalhada de todas as conexões de rede, mostrando o comando, PID, usuário, tipo de dispositivo, nó e nome da porta/endereço IP.

  ■ **Exemplo 2.250** *lsof -i*

- Este comando filtra a saída do lsof para mostrar apenas os processos que estão utilizando a porta "80" (porta padrão para HTTP). É extremamente útil para depurar problemas de conexões ativas no sistema operacional.

  ■ **Exemplo 2.251** *lsof -i :80*

- Combinando o parâmetro "-i tcp" para listar apenas conexões TCP com "-u root" para filtrar pelo usuário "root", este comando mostra todas as conexões TCP abertas pelo usuário root. Deverão ser listadas todas as conexões TCP que foram iniciadas ou estão sendo usadas por processos rodando como usuário "root".

  ■ **Exemplo 2.252** *lsof -i tcp -u root*

2.42 bmon

Descrição 2.42 O utilitário bmon é um monitor de largura de banda de rede que exibe estatísticas em tempo real para todas as interfaces de rede disponíveis. Ele oferece uma interface gráfica baseada em texto (ncurses) que é fácil de usar e fornece informações detalhadas sobre o tráfego de entrada e saída, erros e taxas de pacotes.

- Este comando simplesmente inicia o bmon e exibe uma visão geral de todas as interfaces de rede ativas no sistema, mostrando suas estatísticas em tempo real.

  ■ **Exemplo 2.253** *bmon*

- O parâmetro "-p" (pattern) permite especificar a interface que se deseja monitorar. Neste caso, bmon se concentrará apenas na interface "eth0".

  ■ **Exemplo 2.254** *bmon -p eth0*

3. Procedimentos

Neste capítulo são apresentados procedimentos comuns para configurações de rede no sistema operacional Linux.

3.1 Renomear interface

Para trocar o nome da interface **eth0** para **externo** em tempo de execução

Procedimento 3.1.1

ip link set **eth0** down

ip link set **eth0** name **externo**

ip link set **externo** up

Para trocar o nome da interface **eth0** para **externo** de forma permanente, editar o arquivo **/etc/udev/rules.d/70-persistent-net.rules** e trocar o nome da interface que corresponde ao endereço MAC:

Procedimento 3.1.2

SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:0c:29:1d:86:fd", ATTR{type}=="1", KERNEL=="eth*", NAME="**eth0**"

SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:0c:29:1d:86:fd", ATTR{type}=="1", KERNEL=="eth*", NAME="**externo**"

3.2 Uso de VLANs

Para ativar o uso de VLANs no Linux, deve ser usado o módulo **8021q**. Por exemplo:

Procedimento 3.2.1

modprobe 8021q

Para criar uma interface virtual deve-se utilizar o comando **vconfig**. Por exemplo, para criar a interface eth1.100 com a marcação 100

Procedimento 3.2.2

vconfig add eth1 100

Para evitar problemas com a filtragem de pacotes, deve-se ativar a **flag** para tornar o **dump** (uso em capturas de tráfego de rede) da interface tal como se não houvesse VLANs. Por exemplo:

Procedimento 3.2.3

vconfig set_flag eth1.100 1

Para remover uma VLAN, usa-se o comando **vconfig** com o parâmetro **rem**. Por exemplo, para remover a interface eth1.100:

Procedimento 3.2.4

vconfig rem eth1.100

Para adicionar as marcações de VLANs. No exemplo é adicionada a VLAN ID 100 na interface eth1, criando a interface eth1.100. Para criar múltiplas interfaces marcadas basta repetir o comando variando o name (nome) e o vlan id.

Procedimento 3.2.5 ip link add link eth1 name eth1.100 type vlan id 100

Para remover uma interface marcada para uso de VLANs. No exemplo a remoção da interface eth1.100

Procedimento 3.2.6 ip link delete eth1.100

Para definir um endereço IP de uma interface com VLAN, usa-se o comando **ifconfig** padrão. Por exemplo, para definir o IP 10.0.0.100 na interface eth1.100:

Procedimento 3.2.7

ifconfig eth1.100 10.0.0.100/24

3.3 Interface em Bridge

Para criar uma bridge de nome br0

Procedimento 3.3.1

ip link add br0 type bridge
brctl addbr br0

Para adicionar a interface eth0 na bridge br0

Procedimento 3.3.2

brctl addif br0 eth0
ip link set eth0 master br0

Para mostrar informações sobre a bridge br0

Procedimento 3.3.3

brctl show

Para mostrar a tabela MAC da bridge br0

Procedimento 3.3.4

brctl showmacs br0

Para ativar a bridge de nome br0

Procedimento 3.3.5

ip link set up dev br0

Para desativar a bridge de nome br0

Procedimento 3.3.6

ip link set dev br0 down

Para remover a interface eth0 de uma bridge de nome br0

Procedimento 3.3.7

ip link set eth0 nomaster

ip link set eth0 down

Para remover uma bridge de nome br0

Procedimento 3.3.8

ip link delete br0 type bridge

brctl delbr br0

Para configurar a interface bridge na inicialização

Procedimento 3.3.9

Debian

Arquivo /etc/network/interfaces

auto lo br0

iface lo inet loopback

iface eth0 inet manual

iface eth1 inet manual

Bridge br0

iface br0 inet static

bridge_ports eth0 eth1

address 192.168.200.3

broadcast 192.168.200.255

netmask 255.255.255.0

gateway 192.168.200.1

Procedimento 3.3.10

Centos

Arquivo /etc/sysconfig/network-scripts/ifcfg-br0

DEVICE=br0

TYPE=Bridge

IPADDR=192.168.200.3

NETMASK=255.255.255.0

ONBOOT=yes
BOOTPROTO=none
NM_CONTROLLED=no
DELAY=0

Arquivo /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
TYPE=Ethernet
HWADDR=AA:BB:CC:DD:EE:FF
BOOTPROTO=none
ONBOOT=yes
NM_CONTROLLED=no
BRIDGE=br0

Arquivo /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE=eth1
TYPE=Ethernet
HWADDR=AA:BB:CC:DD:EE:FE
BOOTPROTO=none
ONBOOT=yes
NM_CONTROLLED=no
BRIDGE=br0

Obs.: é necessária a instalação do pacote bridge-utils para o utilitário brctl

3.4 Interface em Bonding

Para definir qual o modo de operação da interface bonding de nome bond0, editar o arquivo **/etc/modprobe.d/bonding.conf**

Procedimento 3.4.1
alias bond0 bonding
options bond0 miimon=80 mode=0
Obs.: mode=1 (Active-Passive), mode=0 (Round-Robin)

Para listar sobre a interface bond0

Procedimento 3.4.2
cat /proc/net/bonding/bond0

Para criar a interface bond0, com as interfaces físicas eth0 e eth1, no modo round-robin

Procedimento 3.4.3

modprobe bonding

ifenslave bond0 eth0 eth1

ip link set bond0 up

Para trocar a interface ativa para eth1, no modo de operação Active-Passive

Procedimento 3.4.4

ifenslave -c bond0 eth1

Para remover a inerface eth0 do bonding

Procedimento 3.4.5

ifenslave -d bond0 eth0

Ativação da interface bond0 na inicialização

Procedimento 3.4.6

CentOS

Arquivo /etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE="eth0"

BOOTPROTO="none"

ONBOOT="yes"

TYPE="Ethernet"

MASTER=bond0

SLAVE=yes

Arquivo /etc/sysconfig/network-scripts/ifcfg-eth1

DEVICE="eth1"

BOOTPROTO="none"

ONBOOT="yes"

TYPE="Ethernet"

MASTER=bond0

SLAVE=yes

Arquivo /etc/sysconfig/network-scripts/ifcfg-bond0

DEVICE=bond0

ONBOOT=yes

BOOTPROTO=static

IPADDR=192.168.200.3

PREFIX=24

NETWORK=192.168.200.0
GATEWAY=192.168.200.1

Procedimento 3.4.7

Debian

Arquivo /etc/network/interfaces

auto bond0
iface bond0 inet static
address 192.168.200.3
netmask 255.255.255.0
network 192.168.200.0
gateway 192.168.200.1
slaves eth0 eth1
bond_mode active-backup
bond_mode 0 para Round-Robin
bond_miimon 100
bond_downdelay 200
bond_updelay 200

3.5 Configurações de interface em arquivos

Configurações de interfaces de rede em sistemas padrão **Debian**.

Procedimento 3.5.1

Arquivo de configuração: /etc/network/interfaces

 # Exemplo de IPv4

auto eth0
iface eth0 inet static
address 192.168.200.3
netmask 255.255.255.0
gateway 192.168.200.254
broadcast 192.168.200.255
dns-nameservers 192.168.200.1 8.8.8.8
dns-search empresa.local

 # Exemplo de IPv6

iface eth0 inet6 static
address fc00:0:2010:60::190

```
netmask 64
gateway fc00:0:2010:60::191

# Exemplo com o uso de VLANs
# Primeira opção: chamar um script para a criação das interfaces marcadas
auto eth1
iface eth1 inet manual
up ifconfig eth1 0.0.0.0 up
up /root/vlans/vlan.sh

# Segunda opção: adicionar as interfaces com a nomeação para uso de VLANs
auto eth0.2
iface eth0.2 inet static
address 192.168.2.1
netmask 255.255.255.0

auto eth0.3
iface eth0.3 inet static
address 192.168.3.1
netmask 255.255.255.0

# IP alias
auto eth0:1
iface eth0:1 inet static
address 192.168.1.7
netmask 255.255.255.0
broadcast 192.168.1.255
network 192.168.1.0
```

Configurações de interfaces de rede em sistemas padrão **Redhat/CentOS)**.

Procedimento 3.5.2

```
Arquivo de configuração: /etc/sysconfig/network-scripts/ifcfg-eth0
# Exemplo de IPv4 e IPv6
DEVICE="eth0"
BOOTPROTO="static"
BROADCAST="192.168.200.255"
DNS1="8.8.8.8"
```

```
GATEWAY="192.168.200.254"
HWADDR="00:50:56:A8:6F:66"
IPADDR="192.168.200.3"
IPV6ADDR="fc00:0:2010:60::116/64"
IPV6INIT="yes"
IPV6_AUTOCONF="no"
NETMASK="255.255.255.0"
NM_CONTROLLED="yes"
ONBOOT="yes"
TYPE="Ethernet"
UUID="849a1bcf-9d10-4fca-a910-6b0e9af18aba"
IPV6_DEFAULTGW=fc00:0:2010:60::191

# VLAN (um arquivo para cada interface)
# Arquivo de configuração: /etc/sysconfig/network-scripts/ifcfg-eth1.192
DEVICE=eth1.192
BOOTPROTO=none
ONBOOT=yes
IPADDR=192.168.1.1
PREFIX=24
NETWORK=192.168.1.0
VLAN=yes

# IP alias
# Arquivo de configuração: /etc/sysconfig/network-scripts/ifcfg-eth1:0
DEVICE=eth1:0
BOOTPROTO=none
ONBOOT=yes
IPADDR=192.168.1.1
PREFIX=24
NETWORK=192.168.1.0
NAME=eth0:0
```

4. Utilitários Avançados

Neste capítulo são apresentados utilitários avançados de redes para administração de sistemas operacionais Linux. Estas ferramentas provêm poderosas funcionalidades que demandam uma base sólida de conhecimentos na área de redes para poderem ser aproveitados da forma correta.

4.1 tcpdump

[1]

Descrição 4.1 Ferramenta para captura de tráfego de rede padrão em sistemas Linux. O Tcpdump é composto por uma ferramenta para interação com o usuário, por uma linguagem de filtros denominada BPF (BSD packet filter) e pela biblioteca Libpcap que possibilita o desenvolvimento de outras ferramentas de captura compatíveis com o formato de arquivos pcap, por exemplo o Wireshark. Parâmetros principais para a realização de capturas:

- "-e" - Mostra informações da camada de enlace (VLAN, endereço físico)
- "-F" - Utiliza um arquivo com os filtros para captura (os filtros em linha de comando serão ignorados)
- "-i" - Define qual interface será utilizada para captura. Por padrão, será escolhida a interface com menor número de identificação. No Linux, pode-se utilizar o parâmetro "any" para capturar em qualquer interface, em modo NÃO promíscuo. Para capturar pacotes na interface de loopback, utilizar a interface "lo".
- "-A" – mostra o conteúdo dos pacotes em formato texto (ASCII).
- "-n" - Não resolve os endereços IP para o nome (DNS). Importante para melhorar o desempenho das capturas e evitar perdas de pacotes.
- "-nn" - Não resolve endereços IP para o nome (DNS) e não converte o número da porta para o nome do serviço (Ex.: porta 80 para http)
- "-vv" e "-vvv" - Modo verbose, mostra mais informações sobre os pacotes. Um "v" a mais, aumenta o nível de detalhamento dos pacotes
- "-w" - Salva a captura em arquivo no formato da biblioteca Libpcap (poderá ser aberto no Wireshark)
- "-s" – Define o tamanho do pacote a ser capturado. Por padrão é 64KB (Exemplos: -s120 (captura os primeiros 120 Bytes de cada pacote, -s0 corresponde ao pacote inteiro)

Procedimento 4.1.1 **Opções:**

Para capturar pacotes na interface padrão, basta executar a ferramenta sem parâmetros

■ **Exemplo 4.1** *tcpdump*

Outros parâmetros são relacionados ao gerenciamento da captura Pacotes da interface eth1, sem resolver o nome dos hosts ou as portas de comunicação e salvar a captura no arquivo /tmp/salvo.cap)

[1] https://www.youtube.com/watch?v=1LUk6fw6W6M

■ **Exemplo 4.2** *tcpdump –i eth1 –nn –w /tmp/salvo.cap*

A filtragem de pacotes é feita por meio de parâmetros da linguagem BPF Pacotes com origem ou destino do IP 10.0.5.10 e que sejam do protocolo UDP)

■ **Exemplo 4.3** *tcpdump host 10.0.5.10 and udp*

Procedimento 4.1.2 Filtros Básicos:

Por endereço IP, origem ou destino

■ **Exemplo 4.4** *tcpdump host 10.0.89.15*

Por endereço IP, como origem dos pacotes

■ **Exemplo 4.5** *tcpdump src host 10.0.89.15*

Por endereço IP, como destino dos pacotes

■ **Exemplo 4.6** *tcpdump dst host 10.0.89.15*

Por porta da camada de transporte, qualquer protocolo

■ **Exemplo 4.7** *tcpdump port 80*

Por porta de origem da camada de transporte, qualquer protocolo

■ **Exemplo 4.8** *tcpdump src port 80*

Por porta de destino da camada de transporte, qualquer protocolo

■ **Exemplo 4.9** *tcpdump dst port 80*

Por porta do protocolo TCP, origem ou destino

■ **Exemplo 4.10** *tcpdump tcp port 80*

Por porta do protocolo TCP, destino

■ **Exemplo 4.11** *tcpdump tcp dst port 80*

Por porta do protocolo TCP, origem

■ **Exemplo 4.12** *tcpdump tcp src port 80*

Por porta do protocolo UDP, origem ou destino

■ **Exemplo 4.13** *tcpdump udp port 53*

Procedimento 4.1.3 Filtros Diversos:

Filtrar todos os pacotes com destino ou origem de determinada rede

■ **Exemplo 4.14** *tcpdump net 192.168.254.0/24*

Filtrar todos os pacotes com destino a determinada rede

■ **Exemplo 4.15** *tcpdump dst net 192.168.254.0/24*

Filtrar todos os pacotes com origem de determinada rede

■ **Exemplo 4.16** *tcpdump src net 192.168.254.0/24*

Filtrar todos os pacotes que não sejam originados ou destinados a determinada rede

■ **Exemplo 4.17** *tcpdump not net 192.168.254.0/24*

Filtrar todos os pacotes do protocolo ICMP

■ **Exemplo 4.18** *tcpdump icmp*

Filtrar todos os pacotes do protocolo ARP

■ **Exemplo 4.19** *tcpdump arp*

Filtrar todos os pacotes do protocolo IPv6

■ **Exemplo 4.20** *tcpdump ip6*

Filtrar todos os pacotes multicast ou broadcast

■ **Exemplo 4.21** *tcpdump multicast or broadcast*

Filtrar pacotes de determinado endereço físico, origem ou destino

■ **Exemplo 4.22** *tcpdump ether host 10:BF:48:89:67:1B*

Filtrar pacotes de determinado endereço físico, origem

■ **Exemplo 4.23** *tcpdump ether src host 10:BF:48:89:67:1B*

Filtrar pacotes de determinado endereço físico, destino

■ **Exemplo 4.24** *tcpdump ether dst host 10:BF:48:89:67:1B*

Filtrar pacotes de determinada vlan

■ **Exemplo 4.25** *tcpdump vlan 200*

Filtrar os pacotes de multicast ou broadcast que não sejam originados do endereço físico 10:BF:48:89:67:1B

■ **Exemplo 4.26** *tcpdump multicast or broadcast and not ether host 10:BF:48:89:67:1B*

Filtrar os pacotes da VLAN com ID 200, que sejam nas portas 80 ou 443 TCP ou na porta 53 UDP

■ **Exemplo 4.27** *tcpdump -nn -i eth1 vlan 200 and tcp port 80 or tcp port 443 or udp port 53*

Filtrar pacotes do IP 192.178.15.17 que sejam com a porta 80 TCP

■ **Exemplo 4.28** *tcpdump -nn -i eth1 tcp port 80 and host 192.178.15.17*

Filtrar pacotes do protocolo ICMP que sejam originados ou destinados aos endereços da rede 192.168.254.0/24

■ **Exemplo 4.29** *tcpdump -nn -i eth1 icmp and net 192.168.254.0/24*

4.2 socat

[2]

Descrição 4.2 Comando que pode funcionar com um repassador de conexões genérico e um cliente de conexões TCP e UDP.

[2] https://www.youtube.com/watch?v=tX4V2yzRmbY

Procedimento 4.2.1 Exemplos de uso:

Redireciona as conexões locais na porta 80 para o IP 192.168.10.15 na porta 8080

- **Exemplo 4.30** *socat TCP-LISTEN:80,fork TCP:192.168.10.15:8080*

  Conecta na porta 80 do endereço IP 10.1.1.1

- **Exemplo 4.31** *socat - TCP:10.1.1.1:80*

4.3 sshpass

[3]

Descrição 4.3 Comando que atua como um cliente não interativo de SSH, possibilitando o uso em scripts e, principalmente, em equipamentos onde não é possível usar autenticação por certificados, tais como switches, APs e roteadores.

Procedimento 4.3.1 Exemplos de uso:

Acessa o servidor de SSH no IP 192.168.254.15 com o usuário "aluno" e a senha "senha123"

- **Exemplo 4.32** *sshpass -p 'senha123' ssh aluno@192.168.254.15*

  Acessa o servidor de SSH no IP 192.168.254.15 com o usuário "aluno" e a senha "senha123" e não checa a chave do host.

- **Exemplo 4.33** *sshpass -p 'senha123' ssh -o StrictHostKeyChecking=no \ aluno@192.168.254.15*

  Acessa o servidor SSH 192.168.10.10 e executa o comando "iwconfig ath0", salvando o resultado em /tmp/saida.txt. A conexão terá o limite de 10s, antes de encerrar por timeout.

- **Exemplo 4.34** *sshpass -p "senha" ssh -o ConnectTimeout=10 \ -o StrictHostKeyChecking=no ubnt@192.168.10.10 "iwconfig ath0" > /tmp/saida.txt*

4.4 Python Simple HTTP Server

[4]

Descrição 4.4 Módulo disponível na linguagem Python que disponibiliza um servidor HTTP simplificado no diretório atual para copiar arquivos de forma facilitada.

Procedimento 4.4.1 Exemplos de uso:

Disponibiliza um servidor HTTP na porta 8000 no diretório atual (Python 2.x).

[3] https://www.youtube.com/watch?v=zr91469aXoY
[4] https://www.youtube.com/watch?v=lPcoHqtzl9w

■ **Exemplo 4.35** *python -m SimpleHTTPServer 8000*

Disponibiliza um servidor HTTP na porta 9500 no diretório atual (Python 3.x).

■ **Exemplo 4.36** *python3 -m http.server 9500*

4.5 comandos r (rcp, rexec, rlogin, and rsh)

[5]

Descrição 4.5 Realizar login, execução de comandos e cópias remotas entre hosts. Os comandos "r" são legados, não são recomendados para uso em produção. Entretanto, podem ser úteis em casos onde existam versões de Unix antigas. Utilizam a porta TCP 514.

Procedimento 4.5.1 Exemplos de uso:

Copia de forma recursiva e mantendo as permissões dos arquivos do diretório local /dados para o diretório /backup no servidor1 (o servidor1 é um nome de host disponível no arquivo /etc/hosts da máquina local).

■ **Exemplo 4.37** *rcp -r -p /dados/ servidor1:backup/*

Realiza a execução remota no servidor de IP 192.168.254.95 como usuário root do comando "uptime".

■ **Exemplo 4.38** *rsh -l root 192.168.254.95 uptime*

4.6 net

[6]

Descrição 4.6 Comandos para realizar a administração remota de servidores Samba ou Microsoft Windows.

Procedimento 4.6.1 Exemplos de uso:

Solicita ao host 192.168.254.17 a lista de serviços disponíveis utilizando o usuário teste123 do domínio nomedomain. Será solicitada a senha.

■ **Exemplo 4.39** *net rpc service list -I 192.168.254.17 -U teste123 –W nomedomain*

Realiza o reboot do host 192.168.254.5 utilizando o usuário administrador e a senha senhaadmin

■ **Exemplo 4.40** *net rpc shutdown -r -I 192.168.254.5 -U administrador%senhaadmin*

Verifica o status do serviço Spooler (impressão) do host 192.168.254.17 utilizando o usuário teste123

■ **Exemplo 4.41** *net rpc service status Spooler -I 192.168.254.17 -U teste123*

[5] https://www.youtube.com/watch?v=GzPepbsXeGM
[6] https://www.youtube.com/watch?v=cE35suY7soE

4.7 snmp*

[7]

Descrição 4.7 Comandos para realizar consultas e alterações em agentes do protocolo SNMP (Simple Network Management Protocol). Dois comandos básicos são o snmpget e o snmpwalk. Para realizar o uso dos comandos é necessário conhecer a MIB (Management Information Base) para saber qual OID (Object Identifier) a ser consultado.

Procedimento 4.7.1 Exemplos de uso:

Acessa o agente SNMP no IP 192.168.254.254, com a comunidade "public", versão "2c" do protocolo e solicita informações sobre o OID "System"

- **Exemplo 4.42** *snmpwalk –c public –v2c 192.168.254.254 system*

  Mostra informações sobre a OID sysContact.0 do agente SNMP em localhost

- **Exemplo 4.43** *snmpget -c public -v2c localhost SNMPv2-MIB::sysContact.0*

4.8 nfs*

[8]

Descrição 4.8 O protocolo NFS (Network File System) tem como objetivo o compartilhamento em rede de área de armazenamento. O protocolo pode utilizar o transporte em UDP e/ou em TCP, sendo que a versão 4 utiliza somente porta 2049 (TCP). O protocolo deverá estar ativado no kernel para funcionar. O serviço no lado cliente é composto pelo serviço do NFS e pelo arquivo **/etc/exports**. O arquivo **/etc/exports** define quais áreas de armazenamento serão compartilhadas e as configurações correspondentes. Exemplos de entradas no arquivo **/etc/exports**:

- Compartilhar o diretório /home para qualquer dentro da faixa 10.10.10.0/24, com direitos de leitura e escrita, com o método assíncrono e evita que usuários conectados no compartilhamento possam ter privilégios de root e aplica o ID do usuário nfsnobody
- **/home 10.10.10.0/24(rw,async,root_squash)**
- Compartilhar o diretório /home/ftp para qualquer endereço IP com direito de leitura e escrita
- **/home/ftp *(rw)**
- Compartilhar o diretório /dados para qualquer IP dentro da faixa 172.26.0.0/16 com direitos de somente leitura.
- **/dados 172.26.0.0/16(ro)**

Após alterar o arquivo **/etc/exports** deve-se executar o comando **exportfs -a** para aplicar as alterações.

[7] https://www.youtube.com/watch?v=O-VI75LsCdA
[8] https://www.youtube.com/watch?v=3oA5LDLIKh8

Procedimento 4.8.1 Montagem de compartilhamentos NFS

Montagem no /home local do /home compartilhado do servidor com o IP 10.0.0.1

■ **Exemplo 4.44** *mount 10.0.0.1:/home /home*

Montagem no /arquivos-remotos local do /dados compartilhado do servidor com o IP 10.0.0.1

■ **Exemplo 4.45** *mount 10.0.0.1:/dados /arquivos-remotos*

Montagem no /backup local do /dados compartilhado do servidor com o IP 10.0.0.1 usando a versão 4 do NFS

■ **Exemplo 4.46** *mount -t nfs4 10.0.0.1:/dados /backup*

Para desmontar o compartilhamento, o comando umount padrão é utilizado. No exemplo está sendo desmontado o /home local

■ **Exemplo 4.47** *umount /home*

Para mostrar os compartilhamentos disponíveis em um servidor com o endereço IP 192.168.67.12

■ **Exemplo 4.48** *showmount -e 192.168.67.12*

Procedimento 4.8.2 Exemplos de configurações do arquivo **/etc/exports**

Diretório: /dados
Endereços IP com acesso: 192.168.254.0/24
Parâmetros:
rw - Liberado acesso de leitura e escrita
sync - Modo síncrono (só após confirmada a operação é feita a escrita em disco), configuração padrão
no_root_squash – ignora qual o usuário remoto, basta ser um dos endereços liberados para ter acesso como root.

■ **Exemplo 4.49** */dados 192.168.254.0/24(rw,sync,no_root_squash)*

Diretório /pub
Endereço: * (qualquer um)
Parâmetros:
ro - somente leitura
insecure – libera portas maiores que 1024 como origem no cliente (usuário remoto não precisa ser root)
all_squash – todos os acessos serão mapeados para um usuário comum, normalmente nobody.

■ **Exemplo 4.50** */pub *(ro,insecure,all_squash)*

Diretório: /pub2
Endereço IP com acesso: 192.168.254.219
Parâmetros:
rw - Liberado acesso de leitura e escrita
all_squash – faz o mapeamento de usuário e grupo para o UID e GID definidos no parâmetros anonuid e anongid. Neste caso, o UID e GID 99 são do usuário nobody.

■ **Exemplo 4.51** */pub2 192.168.254.219(rw,all_squash,anonuid=99,anongid=99)*

4.9 sshfs

[9]

Descrição 4.9 sshfs (SSH Filesystem) se comporta como um sistema de arquivos para uso remoto baseado no protocolo SSH (SFTP) e com suporte da biblioteca FUSE. O sshfs possibilita a montagem de diretórios remotos e o uso tal como fossem locais, de forma simples. Desta forma, simplifica o compartilhamento devido ao usar somente a porta de SSH, com autenticação e criptografia já existentes. Para instalar o serviço:

- Debian - apt install sshfs
- CentOS - yum --enablerepo=powertools install fuse-sshfs

Procedimento 4.9.1

Neste exemplo será montado o "/home/aluno" originado no servidor remoto (endereço IP_Servidor) no diretório local "/arq_remotos".

Para este procedimento funcionar, o diretório "arq_remotos" já deverá existir no cliente local que está executando o comando sshfs.

O usuário "aluno" deverá ter uma conta no servidor remoto por SSH e permissões para acessar o diretório "/home/aluno".

No cliente local deverá estar montado o "/home/aluno" remoto no diretório local "arq_remotos".

■ **Exemplo 4.52** *sshfs aluno@IP_Servidor:/home/aluno /arq_remotos*

4.10 nmap

Descrição 4.10 A ferramenta nmap possibilita a exploração de rede e verificações de segurança, sendo uma das melhores opções para scanner de portas e serviços. A ferramenta possui um repositório de scripts que podem ser usados em atividades de pentest de forma simples e prática.

Procedimento 4.10.1 Um dos usos mais comuns da ferramenta é realizar varreduras de endereços de hosts. Existem diversas formas de definir a faixa de endereços a serem buscados.

Faz uma varredura básica no endereço www.empresa.com.br

■ **Exemplo 4.53** *nmap www.empresa.com.br*

Faz uma varredura básica, em modo verbose, no endereço www.empresa.com.br

■ **Exemplo 4.54** *nmap -v www.empresa.com.br*

Faz uma varredura básica nos endereços 10.0.0.10, 10.0.0.50 e 172.16.7.10

■ **Exemplo 4.55** *nmap 10.0.0.10, 10.0.0.50 e 172.16.7.10*

Faz uma varredura básica nos endereços 192.168.0.0 até 192.168.0.255

■ **Exemplo 4.56** *nmap 192.168.0.**

[9] https://www.youtube.com/watch?v=yDA-Ae0ryaU

■ **Exemplo 4.57** *nmap 192.168.0.0/24*

Faz uma varredura básica nos endereços contidos no arquivo lista.txt. Os endereços devem estar um em cada linha do arquivo.

■ **Exemplo 4.58** *nmap -iL lista.txt*

Faz uma varredura básica nos endereços da faixa 172.26.10.50 até 172.26.10.78

■ **Exemplo 4.59** *nmap 172.26.10.50-78*

Faz uma varredura básica nos endereços 192.168.0.0 até 192.168.0.255, excluindo o endereço 192.168.0.50.

■ **Exemplo 4.60** *nmap 192.168.0.* --exclude 192.168.0.50*

Procedimento 4.10.2 A varredura de portas é outra funcionalidade comum da ferramenta nmap.

Faz uma varredura de portas TCP 80 no endereço www.empresa.com.br

■ **Exemplo 4.61** *nmap -p 80 www.empresa.com.br*

Faz uma varredura de portas TCP 80 e 443 no endereço www.empresa.com.br

■ **Exemplo 4.62** *nmap -p T:80,443 www.empresa.com.br*

Faz uma varredura de portas UDP 53 e 161 no endereço 172.17.90.7

■ **Exemplo 4.63** *nmap -sU 53,161 172.17.90.7*

Faz uma varredura de faixa de portas TCP 1000 a 1500 no endereço 172.17.90.7

■ **Exemplo 4.64** *nmap -p 1000-1500 172.17.90.7*

Faz uma varredura para identificar as versões dos serviços em execução no endereço 172.17.90.7

■ **Exemplo 4.65** *nmap -sV 1000-1500 172.17.90.7*

Faz uma varredura para identificar as 1000 portas mais comuns, com o método TCP Syn (não completa a conexão) no endereço 172.17.90.7

■ **Exemplo 4.66** *nmap -sT 172.17.90.7*

Faz uma varredura para 50 portas mais utilizadas, no endereço 172.17.90.7

■ **Exemplo 4.67** *nmap --top-ports 50 172.17.90.7*

Realiza a varredura em modo stealthy (mínimo ruído gerado) no endereço 10.100.5.1

■ **Exemplo 4.68** *nmap -sS 10.100.5.1*

Procedimento 4.10.3 O uso de funcionalidades específicas para burlar firewalls e detectar vulnerabilidades em sistemas operacionais e aplicações são opções avançadas que o nmap provê e são de importante ajuda nas rotinas de administração de redes.

Para atualizar o repositório de scripts

■ **Exemplo 4.69** *nmap --script-updatedb*

Para executar varredura de vulnerabilidades nos endereços da rede 172.27.1.0/24

■ **Exemplo 4.70** *nmap -v --script vuln 172.27.1.0/24*

Para executar scripts de descoberta de vulnerabilidades no serviço de Samba/-Domínio na porta TCP 445 nos endereços da rede 172.27.1.0/24

■ **Exemplo 4.71** *nmap -p 445 --script smb-os-discovery 172.27.1.0/24*

Para executar testes de força-bruta contra o serviço de SSH (porta TCP 22) no endereço 172.27.1.1. Os parâmetros usuarios.txt e senhas.txt devem ser preenchidos com um item por linha. Caso não sejam usados arquivos de usuários e senhas personalizados, o script usará uma lista padrão.

■ **Exemplo 4.72** *nmap -p 22 --script ssh-brute --script-args \
userdb=usuarios.txt,passdb=senhas.txt --script-args \
ssh-brute.timeout=4s 172.27.1.1*

Realiza a detecção de do sistema operacional e dos serviços em execução (parâmetro "-A") e utiliza o parâmetro "-T4" para execução mais rápida para o endereço 172.27.1.1

■ **Exemplo 4.73** *nmap -A -T4 172.27.1.1*

Para executar testes de força-bruta contra o serviço MS-SQL Server no endereço 172.27.1.1. Os arquivos usuarios.txt e senhas.txt devem ser criados com um item por linha.

■ **Exemplo 4.74** *nmap -p 1433 --script ms-sql-brute \
--script-args userdb=usuarios.txt,passdb=senhas.txt 172.27.1.1*

Realiza varredura para busca de malware no endereço 172.27.1.1

■ **Exemplo 4.75** *nmap -sV --script=http-malware-host 172.27.1.1*

Realiza varredura para busca de malware, com o uso da lista do Google, no endereço 172.27.1.1

■ **Exemplo 4.76** *nmap -p80 --script http-google-malware 172.27.1.1*

Procedimento 4.10.4 Para realizar uma varredura de hosts e portas com endereços IPv6 deve-se usar previamente a detecção dos hosts ativos. Em caso contrário, uma varredura em um prefixo /64 poderia levar centenas de anos para terminar. Uma das formas mais simples é com o uso de shell script com encadeamento de comandos. Outra forma mais elegante é com o script v6disc, disponível em HTTPS://GITHUB.COM/CVMILLER/V6DISC. Procedimento para obter o script v6disc:

- git clone https://github.com/cvmiller/v6disc.git

O script "targets-ipv6-multicast-echo.nse" descobre hosts com endereços IPv6 ativos na rede. A partir desta descoberta poderão ser executados os testes diretamente em cada host. No exemplo, está sendo usada a interface ens33 para poder executar o script.

■ **Exemplo 4.77**
```
ping -6 -c 2 ff02::1|awk '{ print $4 }'|grep ^fe80 \
|awk -F"%" '{ print $1 }'|sort|uniq|xargs -Iativo nmap -v -6 ativo
```

■ **Exemplo 4.78** *v6disc.sh -N*

■ **Exemplo 4.79** *nmap -6 --script=targets-ipv6-multicast-echo.nse \
--script-args 'newtargets,interface=ens33' -sL*

4.11 ngrep

Descrição 4.11 Ferramenta baseada na biblioteca Libpcap, a mesma do Tcpdump, que permite utilizar filtros do Tcpdump e o mecanismos de busca do utilitário **grep** em pacotes de rede. Esta ferramenta pode ser útil para identificar padrões de texto em pacotes e realizar ações.

Procedimento 4.11.1

Neste exemplo serão analisados 60000 pacotes ou um tempo de 30 segundos, o que chegar primeiro, capturados na interface eth1 e registrados no arquivo captura.txt os pacotes que possuam as palavras "GNUTELLA ou Bittorrent" na área de dados.

■ **Exemplo 4.80** *ngrep -d eth1 -i -p -n 60000 -i "GNUTELLA|BitTorrent" *
> captura.txt &PID=$!
sleep 30
kill -1 $PID

4.12 tshark

[10]

Descrição 4.12 O tshark (Terminal Wireshark) foi introduzido como parte do Wireshark para fornecer uma interface de linha de comando. A ferramenta permite que sejam realizadas análises de tráfego de rede sem a necessidade de uma interface gráfica, facilitando tarefas automatizadas e scripts. O funcionamento é similar à ferramenta Tcpdump e possui compatibilidade na sintaxe dos filtros de capturas e no formato de arquivo. Por padrão, o formato de arquivo usado pelo tshark é o pcapng (o formato padrão do Tcpdump é o formato pcap). A instalação da ferramenta está incluída no pacote "wireshark" na maior parte das distribuições. Para saber mais sobre os filtros de amostragem a lista completa está disponível em WIRESHARK DISPLAY FILTER REFERENCE.

Procedimento 4.12.1

Para realizar as capturas, o usuário deverá possuir privilégios de administrador. Ao escolher uma interface para captura, tshark fará a captura de todos os pacotes que entram e saem da interface selecionada. É importante que sejam usados filtros de acordo com o interesse do tráfego de rede.

Lista as interfaces disponíveis para a captura.

■ **Exemplo 4.81** *tshark -D*

Realiza a captura na interface de índice "7"

■ **Exemplo 4.82** *tshark -i 7*

Realiza a captura na interface de nome eth0

■ **Exemplo 4.83** *tshark -i eth0*

Realiza a captura na interface de nome eth0, aplicando o filtro de captura para

[10] https://www.youtube.com/watch?v=qC7WUBznpC4

pacotes que utilizem a porta tcp 80 e possuam o endereço IP 10.0.0.2

■ **Exemplo 4.84** *tshark -i eth0 -f "tcp port 80 and host 10.0.0.2"*

Realiza a captura na interface de índice "7", aplicando o filtro de captura para pacotes que utilizem a porta udp 53 e não possuam o endereço IP 10.0.0.2 como origem ou destino

■ **Exemplo 4.85** *tshark -i 7 -f "udp port 53 and not host 10.0.0.2"*

Realiza a captura por 10 segundos na interface de nome eth0, aplicando o filtro de captura para pacotes que utilizem a porta tcp 22

■ **Exemplo 4.86** *tshark -i eth0 -a duration:10 -f "tcp port 22"*

Realiza a captura de 100 pacotes, sem resolução de nomes (-n), na interface de nome eth0, aplicando o filtro de captura para pacotes que não sejam com endereços de origem ou destino 192.168.1.1

■ **Exemplo 4.87** *tshark -n -c 100 -i 7 -f "not host 192.168.1.1"*

Realiza a captura na interface de índice "7", aplicando o filtro de captura para pacotes que utilizem a porta udp 161 em modo não promíscuo. O padrão do tshark é realizar as capturas em modo promíscuo, ou seja, capturando tráfego de qualquer origem e qualquer destino que seja recebido na interface de rede em captura. Este comportamento é percebido em roteadores e em interfaces que estejam, por exemplo, em modo monitor ou com espelhamento de portas.

■ **Exemplo 4.88** *tshark -i 7 -p "udp port 161"*

Realiza a captura na primeira interface do sistema no formato pcap e salva os pacotes no arquivo "c:\temp\captura.pcap"

■ **Exemplo 4.89** *tshark -F pcap -w c:\temp\captura.pcap*

Realiza a leitura do arquivo "c:\temp\captura.pcap" e aplica o filtro de display para a porta tcp 443. Observar que na análise de capturas deve-se utilizar o formato dos filtros de display e não de captura.

■ **Exemplo 4.90** *tshark -r c:\temp\captura.pcap -Y "tcp.port==443"*

Realiza a leitura do arquivo "c:\temp\captura.pcap" e aplica função de estatísticas para os pacotes do protocolo ICMP.

■ **Exemplo 4.91** *tshark -z icmp,srt -r c:\temp\captura.pcap*

Realiza a captura na interface de índice "7", em modo silencioso e gera estatísticas da hierarquia de protocolos ao encerramento da captura.

■ **Exemplo 4.92** *tshark -i 7 -qz io,phs*

Realiza a captura na interface de índice "7", em modo silencioso e gera estatísticas do protocolo DNS ao encerramento da captura.

■ **Exemplo 4.93** *tshark -z dns,tree -q -i 7*

Realiza a captura na interface de índice "7", em modo silencioso e gera estatísticas do protocolo TCP para os pacotes com a porta 443 ao encerramento da captura.

■ **Exemplo 4.94** *tshark -z conv,tcp -q -i 7 -f "tcp port 443"*

O utilitário **capinfos**, que faz parte do pacote da ferramenta Wireshark, mostra

informações e estatísticas de um arquivo de capturas. No exemplo, o capinfos realiza a leitura do arquivo "c:\temp\captura.pcap" e mostra informações e estatísticas.

■ **Exemplo 4.95** *capinfos c:\temp\captura.pcap*

Lê o arquivo "c:\temp\captura.pcap" e filtra as requisições HTTP GET, escrevendo a saída no arquivo HTTP-GET.pcap

■ **Exemplo 4.96** *tshark -2 -R "http.request.method==GET" -r c:\temp\captura.pcap \ -w HTTP-GET.pcap*

Procedimento 4.12.2

O uso de filtros é parte fundamental para a análise de tráfego e de utilização da ferramenta tshark. Os filtros podem ser usados na captura, denominados de "capture filters" os quais possuem sintaxe igual ao tcpdump, e os filtros de amostragem, denominados de "display filters" que possuem um grande número de protocolos e funcionalidades. Outra funcionalidade útil é o uso de campos específicos dos protocolos que podem ser utilizados para a análise dos pacotes. Qualquer campo do cabeçalho e dos dados podem ser usados como parte do filtro de amostragem.

Para listar todos os campos que podem ser utilizados com os parâmetros "-T fields -e campo". Para saber os campos de interesse com maior facilidade, pode-se usar a ferramenta Wireshark e copiar o nome do campo na interface gráfica.

■ **Exemplo 4.97** *tshark -G fields*

Captura pacotes na interface de índice "7" com porta de destino 53, sem resolução de nomes, apresenta os campos: frame.time (tempo de captura do pacote), ip.src (ip de origem), ip.dst (ip de destino) e dns.qry.name (o conteúdo da query realizada ao servidor de DNS)

■ **Exemplo 4.98** *tshark -i 7 -f "dst port 53" -n -T fields -e frame.time -e ip.src -e ip.dst \ -e dns.qry.name*

Captura pacotes na interface de índice "7" que sejam requisições HTTP (-Y http.request) e mostra os campos http.host (IP ou domínio do servidor) e http.user_agent (versão do cliente de HTTP.

■ **Exemplo 4.99** *tshark -i 7 -Y http.request -T fields -e http.host -e http.user_agent*

Captura pacotes na interface de índice "7", filtra por pacote TCP na porta 443, mostra os campos frame.time (momento da captura do pacote), ip.src (ip de origem) e ip.dst (ip de destino) separados pelo caracter "|", que sejam requisições HTTP (-Y http.request) e mostra os campos http.host (IP ou domínio do servidor) e http.user_agent (versão do cliente de HTTP.

■ **Exemplo 4.100** *tshark -i 7 -n -Y "tcp.port==443" -T fields -E header=y -E separator="|" \ -e frame.time -e ip.src -e ip.dst*

Procedimento 4.12.3

Exemplos de uso da ferramenta Tshark para análise de tráfego e identificação do comportamento de de anomalias dos fluxos de rede.

Captura tráfego na interface de índice "7" e filtra os nomes dos hosts informados pelos certificados (ssl.handshake.type==11) e mostra os campos ip de origem, porta de destino e informações sobre os domínios no certificado.

■ **Exemplo 4.101** *tshark -i 7 -Y "ssl.handshake.type==11" -T fields -e ip.src \ -e tcp.dstport -e x509sat.uTF8String -e x509ce.dNSName*

Captura tráfego na interface de índice "7" e filtra os nomes dos hosts informados pelos certificados (ssl.handshake.type==11) e mostra os campos ip de origem, porta de destino e informações sobre os domínios no certificado.

■ **Exemplo 4.102** *tshark -i 7 -Y "ssl.handshake.type==11" -T fields -e ip.src -e tcp.dstport \ -e x509sat.uTF8String -e x509ce.dNSName*

Captura tráfego na interface de índice "7", filtra a porta UDP 53 (DNS) e mostra os endereços IP destino (endereços dos servidores de DNS sendo usados).

■ **Exemplo 4.103** *tshark -i 7 -q -Y "udp.dstport == 53" -T fields -e ip.dst*

Captura tráfego na interface de índice "7", filtra o tráfego do protocolo DNS e mostra ao final da captura o número de respostas por segundo para o servidor de IP 8.8.8.8

■ **Exemplo 4.104** *tshark -i 7 -q -z "io,stat,1,COUNT(dns.flags.rcode)dns.flags.rcode and ip.addr==8.8.8.8"*

■ **Exemplo 4.105** *tshark -i 7 -q -Y "ip.addr==8.8.8.8" \ -z "io,stat,1,COUNT(dns.flags.rcode)dns.flags.rcode"*

Captura tráfego na interface de índice "7", filtra os pacotes que tenham destino para o endereço 192.168.254.101 e mostra ao final estatísticas de cada fluxo encontrado.

■ **Exemplo 4.106** *tshark -i 7 -q -z conv,ip -Y "ip.dst==192.168.254.101"*

Captura tráfego na interface de índice "7", filtra os pacotes que tenham destino para o endereço 192.168.254.101 na porta TCP 443 e mostra ao final estatísticas de cada fluxo encontrado.

■ **Exemplo 4.107** *tshark -i 7 -q -z conv,ip -Y "ip.dst==192.168.254.101 and tcp.port==443"*

Captura tráfego na interface de índice "7", filtra os pacotes que tenham origem ou destino para o endereço IPv6 2804:0:2010:160::8 e mostra ao final estatísticas de cada endereço que fez comunicação com o endereço do filtro.

■ **Exemplo 4.108** *tshark -i 7 -q -n -z endpoints,ipv6,"ipv6.addr==2804:0:2010:160::8"*

Captura tráfego na interface de índice "7", filtra os pacotes que tenham origem ou destino para o endereços IPv4 10.10.1.15 e 10.0.0.1 e mostra ao final estatísticas dos fluxos TCP gerados entre os hosts

■ **Exemplo 4.109** *tshark -i 7 -q -n -z flow,tcp,network,"ip.addr==10.10.1.15 and \ ip.addr==10.0.0.1"*

Captura tráfego na interface de índice "7", filtra os pacotes que tenham origem ou destino para a porta TCP 30000 e mostra ao final estatísticas dos hosts e dos fluxos TCP encontrados.

■ **Exemplo 4.110** *tshark -i 7 -n -qz conv,tcp,"tcp.port==30000" -qz conv,ip,"tcp.port==30000"*

Procedimento 4.12.4

Exemplos **avançados** de uso da ferramenta Tshark para análise de tráfego e identificação do comportamento de anomalias dos fluxos de rede.

Captura tráfego na interface de índice "7" e filtra os pacotes com uso da porta TCP 443, com o endereço IP 10.0.0.1 e que sejam retransmissões TCP.

■ **Exemplo 4.111** *tshark -i 7 -n -Y "tcp.analysis.retransmission and tcp.port==443 and \ ip.addr==10.0.0.1"*

Captura tráfego na interface de índice "7" e filtra os pacotes do protocolo rtp (usado em ligações VoIP com o protocolo SIP) e mostra estatísticas ao final da captura.

■ **Exemplo 4.112** *tshark -i 7 -q -n -Y "rtp" -z rtp,streams*

Captura tráfego na interface de índice "7" e filtra os pacotes do protocolo TCP com o flag de Reset ativado (indica final de conexão).

■ **Exemplo 4.113** *tshark -i 7 -n -q -Y "tcp.flags.reset == 1"*

Captura tráfego na interface de índice "7" e filtra os pacotes ICMP com tempo de resposta maior do que 50ms.

■ **Exemplo 4.114** *tshark -i 7 -n -Y "icmp && icmp.resptime > 50"*

Lê o arquivo captura.pcap, filtra pela porta TCP 3306 e para os pacotes com requisições e respostas do protocolo MySQL. O tempo entre os pacotes é mostrado em delta (diferença do tempo entre a requisição e a resposta).

■ **Exemplo 4.115** *tshark -i 7 -t d -n -Y "tcp.port==3306 and mysql.request or \ mysql.response_code" -r c:\temp\captura.pcap*

Captura tráfego na interface de índice "7" e filtra os pacotes com a flag de query lenta ativada.

■ **Exemplo 4.116** *tshark -i 7 -n -Y "mysql.stat.query_was_slow==1"*

Captura tráfego na interface de índice "7" e filtra os pacotes com requisições do protocolo Postgresql, fazendo o despejo em formato ASCII (texto) do conteúdo dos pacotes.

■ **Exemplo 4.117** *tshark -i 7 -n -Y "pgsql.query" --hexdump ascii*

4.13 nmcli

Descrição 4.13 O utilitário em linha de comando nmcli (*Network Manager Command Line Interface*) faz parte do serviço Network Manager para gerenciamento de conexões de rede. Este comando tem o objetivo de realizar configurações básicas e avançadas em conexões de rede.

Procedimento 4.13.1

Exemplos **básicos** de uso da ferramenta nmcli administração de conexões de rede.

Mostra todas as interfaces de rede e suas configurações.

■ **Exemplo 4.118** *nmcli*

Mostra todas as interfaces de rede e suas configurações, com um formato em campos separados.

■ **Exemplo 4.119** *nmcli device show*

Mostra as conexões ativas.

■ **Exemplo 4.120** *nmcli connection show*

Mostra somente as conexões ativas.

■ **Exemplo 4.121** *nmcli connection show --active*

Mostra os estados das conexões e apresenta em modo de múltiplas linhas.

■ **Exemplo 4.122** *nmcli -m t dev status*

Mostra os estados das conexões e apresenta em modo "terse" (em uma mesma linha com os campos separados por ":").

■ **Exemplo 4.123** *nmcli -t dev status*

Mostra informações sobre servidores de DNS ativados nas conexões em IPv4.

■ **Exemplo 4.124** *nmcli -f IP4.DNS device show*

Mostra o nome da interface de rede e informações sobre servidores de DNS ativados nas conexões em IPv4.

■ **Exemplo 4.125** *nmcli -f GENERAL.DEVICE,IP4.DNS device show*

Mostra, de forma reduzida, o nome da interface de rede e informações sobre servidores de DNS ativados nas conexões em IPv4.

■ **Exemplo 4.126** *nmcli -g GENERAL.DEVICE,IP4.DNS device show*

Mostra, de forma reduzida, o nome da interface de rede, endereço IPv4 e endereço MAC nas conexões em IPv4.

■ **Exemplo 4.127** *nmcli -g GENERAL.DEVICE,IP4.ADDRESS,GENERAL.HWADDR \ device show*

Mostra, de forma reduzida e não adicionando o caracter de escape "/", o nome da interface de rede, endereço IPv4 e endereço MAC nas conexões em IPv4.

■ **Exemplo 4.128** *nmcli -e no -g GENERAL.DEVICE,IP4.ADDRESS,GENERAL.HWADDR \
device show*

Mostra não adicionando o caracter de escape "/", o nome da interface de rede, endereço IPv4 e endereço MAC nas conexões em IPv4.

■ **Exemplo 4.129** *nmcli --mode multiline -e no -g GENERAL.DEVICE,IP4.ADDRESS, \ GENERAL.HWADDR device show*

Ativa uma conexão de nome ens160.

■ **Exemplo 4.130** *nmcli con up id ens160*

Desconecta uma conexão de nome ens160, só poderá ser ativada de forma manual.

■ **Exemplo 4.131** *nmcli dev disconnect ens160*

Desconecta uma conexão de nome ens160, sem evitar que a autoreconexão possa atuar.

■ **Exemplo 4.132** *nmcli connection down ens160*

Remove a conexão pelo UID.

■ **Exemplo 4.133** *nmcli con delete c8205dc8-07eb-4c60-82c2-e9410061256a*

Realiza a troca do nome do host para "meuservidor".

■ **Exemplo 4.134** *nmcli general hostname meuservidor*

Cria a conexão vlan-ens224.100 para uso com o VLAN ID 100 e o IPv4 192.168.100.1

■ **Exemplo 4.135** *nmcli con add type vlan con-name vlan-ens224.100 ifname ens224.100 \ dev ens224 id 100 ip4 192.168.100.1/24*

Cria uma conexão "Trabalho" na interface ens224 com IPv4 e IPv6 de forma manual.

■ **Exemplo 4.136** *nmcli con add type ethernet con-name Trabalho ifname ens224 \ ip4 172.19.10.54/24 gw4 172.19.10.254 ip6 2001:0db8:160::100/64 \ gw6 2001:0db8:160::1*

Cria uma conexão "Trabalho2" na interface ens224 com IPv4 em DHCP e IPv6 de forma manual.

■ **Exemplo 4.137** *nmcli con add type ethernet con-name Trabalho2 ifname ens224 \ ipv4.method auto ip6 2001:0db8:161::100/64 gw6 2001:0db8:161::1*

Lista as redes (SSID) percebidas pela interface wl133.

■ **Exemplo 4.138** *nmcli dev wl133 list*

Procedimento 4.13.2

Exemplos **avançados** de uso da ferramenta nmcli administração de conexões de rede.

Lista as redes (SSID) percebidas pela interface wl133.

■ **Exemplo 4.139** *nmcli dev wl133 list*

Adiciona um servidor DNS em uma conexão ens160.

■ **Exemplo 4.140** *nmcli con mod ens160 +ipv4.dns 8.8.4.4*

Verifica se foi aplicado corretamente.

■ **Exemplo 4.141** *nmcli device show ens160*

Remove o servidor 8.8.4.4 na conexão ens160.

■ **Exemplo 4.142** *nmcli con mod ens160 -ipv4.dns 8.8.4.4*

Adiciona uma nova conexão em uma interface desativada de nome ens224.

■ **Exemplo 4.143** *nmcli con add connection.interface-name ens224 type ethernet*

Cria uma nova conexão na interface ens224 com o nome "Faculdade" e define o uso de DHCP para a conexão (padrão).

■ **Exemplo 4.144** *nmcli connection add type ethernet con-name Faculdade ifname ens224*

■ **Exemplo 4.145** *nmcli con mod Faculdade ipv4.method auto*

Cria a conexão Faculdade e adiciona um IPv4 192.168.254.89/24 e o gateway 192.168.254.254.

■ **Exemplo 4.146** *nmcli con add type ethernet con-name Faculdade ifname ens224 ip4 \ 192.168.254.89/24 gw4 192.168.254.254*

Configura os servidores de DNS 8.8.8.8 e 8.8.4.4 na conexão Faculdade.

■ **Exemplo 4.147** *nmcli con mod Faculdade +ipv4.dns "8.8.8.8 8.8.4.4"*

Adiciona o IPv4 172.16.0.1/24 na conexão Faculdade.

■ **Exemplo 4.148** *nmcli con mod Faculdade +ipv4.addresses 172.16.0.1/24*

Modifica a conexão Faculdade para uso de IPv4 manual.

■ **Exemplo 4.149** *nmcli con mod Faculdade ipv4.method manual ipv4.address \ 172.18.10.4/24 ipv4.gateway 172.18.10.1*

Ignora o uso de IPv6 na conexão Faculdade.

■ **Exemplo 4.150** *nmcli con mod Faculdade ipv6.method ignore*

Ativa a conexão Faculdade.

■ **Exemplo 4.151** *nmcli con up Faculdade*

Utiliza o **modo interativo de edição do nmcli** para a conexão de nome "ethernet-ens224"

■ **Exemplo 4.152** *nmcli con edit ethernet-ens224*

Mostra informações das configurações da conexão

■ **Exemplo 4.153** ***nmcli>*** *print*

Adiciona o servidor de DNS 8.8.4.4 e 1.1.1.1

■ **Exemplo 4.154** ***nmcli>*** *set ipv4.dns 8.8.4.4 1.1.1.1*

Remove o servidor de DNS 8.8.4.4

■ **Exemplo 4.155** ***nmcli>*** *remove ipv4.dns 8.8.4.4*

Adiciona o endereço IPv4 10.0.0.1/24 na conexão

■ **Exemplo 4.156** ***nmcli>*** *set ipv4.addresses 10.0.0.1/24*

Salva as configurações de forma temporária

■ **Exemplo 4.157** ***nmcli>*** *save temporary*

Salva as configurações de forma permanente

■ **Exemplo 4.158** ***nmcli>*** *save persistent*

4.14 iscsi*

[11]

[11] https://www.youtube.com/watch?v=w0kDt9CfOyE

Descrição 4.14 O protocolo iSCSI (Internet Small Computer Systems Interface), publicado na RFC 3270 (https://tools.ietf.org/html/rfc3720), que tem como objetivo disponibilizar armazenamento em disco de forma remota. O protocolo tem como princípio utilizar o padrão para conexões de periféricos SCSI de forma remota, fazendo com que o armazenamento disponibilizado em rede seja utilizado tal como fosse um disco local. Este protocolo é o padrão de fato para disponibilização de armazenamento em hospedeiros para virtualização. O protocolo iSCSI tem como objetivo utilizar o padrão SCSI em rede, "enganando" os hosts para parecer que o disco está local. O disco remoto ficará disponível na máquina tal como se fosse um disco local. Poderá ser formatado e montado da forma que for necessário O iSCSI não se comporta como um compartilhamento de discos. O comportamento é diferente do NFS, onde os dados são apresentados ao sistema remoto de forma idêntica ao local onde estão armazenados.

Os componentes da arquitetura do protocolo iSCSI são:

- **Initiator**: É o nome do cliente iSCSI. Possui acesso aos dispositivos em nível de bloco, normalmente um dispositivo de armazenamento. Um cliente poderá ter acesso a diversos dispositivos iSCSI
- **Target**: É o nome dado ao bloco de armazenamento disponível no servidor iSCSI, o qual fornece o dispositivo a ser utilizado (normalmente disco). O servidor tem o nome de Portal e pode possuir vários IPs de acesso Um dispositivo deve ser acessado por APENAS um cliente. Se mais de um usar haverá perdas de dados!

O protocolo iSCSI define o nome dos recursos em duas partes:

- Type (Tipo)
- Unique Name (Nome Único)

Os tipos podem ser:

- iqn. : nome qualificado iscsi (iscsi qualifiled name)
- eui. : identificador eui-64 bit

Exemplo: iqn.2020-04.local.empresa:storage01.disk01-10g

Obs.: o nome do **target** obedece ao padrão IQN (*iSCSI Qualified Name*) que determina o seguinte formato: **iqn.ano-mês.local.meudomínio:descriçãodotarget**

O protocolo iSCSI utiliza TCP e a porta 3260 por padrão. O uso do conceito de portal possibilita o uso de múltiplos caminhos para realizar balanceamento e redundância de acesso aos targets. O protocolo é baseado em mensagens de controle e comandos SCSI encapsulados para acesso aos targets.

Procedimento 4.14.1

Neste exemplo serão realizados os procedimentos para instalação do portal iSCSI em uma distribuição CentOS.
Procedimentos:

Instalação: **yum install scsi-target-utils**

Criação de um arquivo para servidor como área a ser disponibilizada para uso. Escolher um disco, partição ou arquivo para ser um target.

Exemplo 4.159

```
Criação de um arquivo de 40GB:
dd if=/dev/zero of=/discos/disco01.img count=0 bs=1 seek=40G
```

Escolher um nome de acordo com o padrão iqn.

■ **Exemplo 4.160**

```
Exemplo de nome de target:
iqn.2020-05.local.empresa:dellstorage.disk01-40G
```

Disponibilizar o espaço escolhido como target. Adicionar no arquivo **/etc/tgt/targets.conf**

■ **Exemplo 4.161**

```
<target iqn.2020-05.local.empresa:dellstorage.disk01-40G>

   backing-store /discos/disk01.img

</target>
```

Para adicionar uma partição /dev/sdc1

■ **Exemplo 4.162**

```
<target iqn.2019-03.local.servidor1:tiaomacale2018.sdc1>
    backing-store /dev/sdc1
</target>
```

Para adicionar um disco inteiro (/dev/sdd), limitando o IP do cliente (initiator) e com autenticação de usuário e senha

■ **Exemplo 4.163**

```
<target iqn.2019-03.local.servidor1:tiaomacale2018.sdd>
    backing-store /dev/sdd
     initiator-address 192.168.200.3
     incominguser aluno senac2010
</target>
```

Fazer um reload no serviço

■ **Exemplo 4.164**

```
/etc/init.d/tgtd reload
```

Para verificar se o target está disponível

■ **Exemplo 4.165**

```
tgtadm --mode target --op show
```

Procedimento 4.14.2

Neste exemplo serão realizados os procedimentos para instalação do initiator iSCSI (cliente) em uma distribuição CentOS.
Procedimentos:

Instalação: **yum install yum install iscsi-initiator-utils**

Para realizar a descoberta de targets no portal 192.168.200.174

■ **Exemplo 4.166**

```
iscsiadm -m discovery -t st -p 192.168.200.174
```

Para realizar a descoberta de informações sobre determinado target

■ **Exemplo 4.167**

```
iscsiadm -m node -T iqn.2019-03.local.servidor1:tiaomacale2018.sdd \
-p 192.168.200.174
```

Para obter acesso e disponibilizar o target no host local

■ **Exemplo 4.168**

```
iscsiadm -m node -T iqn.2019-03.local.servidor1:tiaomacale2018.sdd \
-p 192.168.200.174 -l
```

Para disponibilizar o target como um disco local

■ **Exemplo 4.169**

```
* Lista os discos e partições disponíveis
** fdisk -l
* Identificar o disco local que aponta para o target iSCSI e abrir com o fdisk
** fdisk /dev/sdb
* Criar uma partição primária e formatar com ext4
** mkfs.ext4 /dev/sdb1
* Descobrir o UUID da nova partição
** blkid /dev/sdb1
* Para montar o disco na inicialização do host local, editar o arquivo
/etc/fstab:
** UUID=d3c7e77b-2217-4610-b54d-38c5d8c6a1d6  /mnt/teste   ext4  _netdev  0 0
```

Para realizar a desconexão do target

■ **Exemplo 4.170**

```
iscsiadm -m node -T iqn.2019-03.local.servidor1:tiaomacale2018.sdd \
-p 192.168.200.174 -u
```

Para remover o target do host local

■ **Exemplo 4.171**

```
iscsiadm -m node -T iqn.2019-03.local.servidor1:tiaomacale2018.sdd \
-p 192.168.200.174 --op delete
```

4.15 sysctl - parâmetros de rede

[12]

Descrição 4.15 O utilitário sysctl permite editar os parâmetros de configuração do kernel do Linux. Desta forma, é possível realizar alterações no comportamento do sistema operacional Linux em rede. O arquivo principal de configurações dos parâmetros está localizado em /etc/sysctl.conf. As alterações nos parâmetros devem ser feitas com cuidado porque podem causar efeitos indesejados. Uma fonte importante para definir os valores dos parâmetros está

[12] https://www.youtube.com/watch?v=6Mro91cJ7fw

baseada nas documentações dos serviços e aplicações que serão disponibilizadas no sistema operacional. Outra consideração crítica é sobre os recursos de hardware que devem estar compatíveis com as alterações nos parâmetros.

Procedimento 4.15.1

São apresentadas as operações básicas para o uso do comando do sysctl.

■ **Exemplo 4.172** Lista todos os parâmetros do kernel:

```
sysctl -a
```

■ **Exemplo 4.173** Aplica os parâmetros do kernel definidos no arquivo /etc/sysctl.conf

```
sysctl -p /etc/sysctl.conf
```

■ **Exemplo 4.174** Mostra a configuração atual do parâmetro do kernel net.ipv4.ip_forward

```
sysctl net.ipv4.ip_forward
```

■ **Exemplo 4.175** Altera a configuração do parâmetro do kernel net.ipv4.ip_forward para "1" que tem como objetivo ativar o roteamento

```
sysctl net.ipv4.ip_forward=1
```

Procedimento 4.15.2

São apresentados parâmetros relacionados às funcionalidades de redes que alteram o comportamento padrão do kernel em sistemas operacionais Linux. Estes parâmetros podem ser executados em linha de comando. Para deixar de forma permanente na próxima inicialização, deve-se incluir o parâmetro no arquivo /etc/sysctl.conf

■ **Exemplo 4.176** net.ipv4.conf.all.accept_redirects:
Controla se a interface aceitará mensagens ICMP Redirect para IPv4. Desabilitar pode aumentar a segurança, prevenindo ataques que tentam redirecionar o tráfego.

```
net.ipv4.conf.all.accept_redirects = 0 (Não aceitar redirecionamentos)
```

■ **Exemplo 4.177** net.ipv4.tcp_window_scaling:
Habilita ou desabilita a extensão de "Window Scaling" para TCP. Esta extensão permite o uso de janelas de recepção TCP maiores que 64KB, o que é crucial para redes de alta largura de banda e alta latência, melhorando o throughput

```
net.ipv4.tcp_window_scaling = 1 (Habilitado)
```

■ **Exemplo 4.178** net.ipv4.tcp_max_syn_backlog:
Define o tamanho máximo da fila de requisições SYN pendentes (conexões que ainda não completaram o handshake de três vias). Um valor maior pode ajudar a prevenir ataques de SYN flood e melhorar o desempenho em servidores com

```
 alta carga  de conexão.

net.ipv4.tcp_max_syn_backlog = 3240000 (3.240.000 conexões pendentes)
```

■ **Exemplo 4.179** `fs.file-max:`

```
  Este parâmetro define o número máximo de arquivos (file handles) que o
  kernel    pode manter   abertos simultaneamente em todo o sistema. Um
  valor alto permite   que mais processos abram   mais arquivos, o que é
  crucial para servidores com   muitas conexões ou aplicações que lidam
  com um grande número de arquivos.

  fs.file-max = 400000 (Permite até 400.000 descritores de arquivo abertos)
```

■ **Exemplo 4.180** `net.ipv4.neigh.default.gc_thresh1:`

```
    O número mínimo de entradas de cache de vizinhos (ARP) a serem mantidas
    pelo kernel. O processo de "garbage collection" (coleta de lixo) para
    entradas de cache só será acionado se o número de entradas exceder este
    valor.

 net.ipv4.neigh.default.gc_thresh1 = 4096 (Mínimo de 4096 entradas)
```

■ **Exemplo 4.181** `net.netfilter.nf_conntrack_max:`

```
 O número máximo de entradas de conexão que a tabela de rastreamento de
 conexões   (conntrack) do Netfilter pode manter. Esta tabela armazena
 informações sobre as  conexões de rede ativas. Um valor baixo pode levar
 a quedas de conexão ou  rejeições de novas conexões em sistemas com alto
 volume de tráfego.

 net.netfilter.nf_conntrack_max = 262144 (262.144 conexões rastreadas)
```

4.16 hping3

Descrição 4.16 O utilitário hping3 é um scanner de porta e um gerador de pacotes de rede versátil, baseado no conceito do ping, mas com muito mais funcionalidades. Ele pode ser usado para testes de firewall, rastreamento de portas, engenharia reversa de pacotes TCP/IP, teste de desempenho de rede e detecção de rootkits. É uma ferramenta poderosa para testes de segurança e depuração de rede.

Procedimento 4.16.1

São apresentados comandos úteis utilizando a ferramenta hping3 para a administração de sistemas e serviços.

■ **Exemplo 4.182** `Envia pacotes SYN para a porta 80 do host "www.empresa.com.br"`

```
O parâmetro "-S" define a flag SYN no pacote, e "-p 80"
especifica a porta de destino. Se a porta estiver aberta, será
recebebido um pacote SYN-ACK.

hping3 -S -p 80 www.empresa.com.br
```

■ **Exemplo 4.183** `O parâmetro "--icmp" instrui o hping3 a enviar pacotes ICMP`

```
echo (como o ping tradicional), e "-c 5" limita o número de
```

```
pacotes a 5. É útil para testar a conectividade básica com um
host. No exemplo serão enviados 5 pacotes de ICMP Echo Request
para o host 192.168.1.1

hping3 --icmp -c 5 192.168.1.1
```

■ **Exemplo 4.184**

```
Este comando gera um grande volume de tráfego SYN e pode
causar uma negação de serviço. O parâmetro "-S" envia pacotes SYN,
"-p 80" direciona para a porta 80, e "--flood" envia pacotes o mais
rápido possível, sem mostrar as respostas. É uma simulação de um
ataque SYN flood.

hping3 -S -p 80 --flood 192.168.1.100
```

■ **Exemplo 4.185**

```
Envia pacotes SYN (-S) para a porta 80 (-p 80) do host
www.empresa.com.br. O parâmetro "-d 100" adiciona 100 bytes
de dados aleatórios ao pacote TCP, o que pode ser útil para
testar a capacidade de processamento de firewalls ou sistemas
de detecção de intrusão. O "--ttl 64" define o Time To Live (TTL)
do pacote para 64, simulando um pacote originado de um sistema Linux comum.

hping3 -S -p 80 -d 100 --ttl 64 www.empresa.com.br
```

■ **Exemplo 4.186**

```
O parâmetro "-A" envia pacotes TCP com a flag ACK definida.
 Um ACK scan é  frequentemente usado para mapear as regras de um firewall:
    * Se houver o recebimento de um RST (Reset), a porta provavelmente não
    está filtrada (está aberta ou fechada sem um firewall bloqueando).
    * Se não houver nenhuma resposta ou uma resposta de
    ICMP "Destination Administratively Prohibited", a porta
    está provavelmente filtrada por um firewall.
    No exemplo, o teste é para a porta 443 (HTTPS) do host 192.168.1.1.

hping3 -A -p 443 192.168.1.1
```

■ **Exemplo 4.187**

```
Este comando envia 10 pacotes UDP ("--udp") para a porta 53
 ("-p 53",  porta padrão para DNS) do servidor DNS do Google (8.8.8.8).
 O "-c 10" limita  o número de pacotes a 10. Isso é útil para testar a
 conectividade UDP para serviços  específicos, como servidores DNS ou NTP.

hping3 --udp -p 53 -c 10 8.8.8.8
```

4.17 ip tunnel

Descrição 4.17 O comando ip tunnel faz parte do conjunto de ferramentas iproute2 e é utilizado para gerenciar interfaces de túnel no kernel Linux. Os túneis permitem encapsular pacotes de um protocolo dentro de pacotes de outro, sendo o princípio básico de VPNs (Virtual Private Network). Embora não forneça criptografia por si só (o que é feito por ferramentas como IPsec), ele é a base para a criação de VPNs simples (como as baseadas em GRE) ou para interconectar redes isoladas com diferentes protocolos, por exemplo IPv4 sobre IPv6 ou IPv6 sobre IPv4. Existem diferentes modos de túnel que podem ser criados com ip tunnel:

* IPIP (IP over IP): O mais simples, encapsula pacotes IPv4 dentro de outros pacotes IPv4. Não suporta multicast.

* GRE (Generic Routing Encapsulation): Mais versátil que IPIP, permite encapsular uma variedade de protocolos de Camada 3 (incluindo multicast) dentro de IP. É amplamente suportado por roteadores Cisco e outros equipamentos de rede.

* SIT (Simple Internet Transition): Originalmente projetado para túneis IPv6 sobre IPv4, mas também pode encapsular IPv4 sobre IPv4.

* VTI (Virtual Tunnel Interface): Projetado para ser usado em conjunto com IPsec, permitindo a integração de túneis com criptografia e autenticação.

Serão apresentados nos exemplos túneis nos modos IPIP e GRE. Para os cenários dos exemplos serão considerados dois servidores com o endereçamento disponível na Tabela 4.1.

Table 4.1: Endereçamento do Cenário de Exemplos do comando ip tunnel

Servidor	IP Público	Rede Interna	IP do Túnel
A	203.0.113.1	10.0.1.0/24	192.168.100.1/30
B	198.51.100.1	10.0.2.0/24	192.168.100.2/30

Procedimento 4.17.1

Será realizada a configuração de um túnel do modo IPIP (IP over IP) entre os servidores A e B que possibilitará a comunicação entre as redes internas.

■ **Exemplo 4.188** `Configuração o túnel IPIP (IP over IP) no Servidor A`

```
Comandos:

ip tunnel add tunelab mode ipip remote 198.51.100.1 local 203.0.113.1 ttl 64
ip addr add 192.168.100.1/30 dev tunelab
ip link set tunelab up
ip route add 10.0.2.0/24 via 192.168.100.2 dev tunelab

Explicação das linhas de comandos:
* "ip tunnel add tunelab mode ipip":
  ** Cria uma nova interface de túnel chamada ‘tunelab‘ no modo IPIP.
* "remote 198.51.100.1":
  ** Define o endereço IP público do Servidor B como a extremidade remota
  do túnel.
* "local 203.0.113.1":
  ** Define o endereço IP público do Servidor A como a extremidade local
  do túnel.
* "ttl 64":
  ** Define o TTL (Time To Live) dos pacotes encapsulados.
* "ip addr add 192.168.100.1/30 dev tunelab":
  ** Atribui um endereço IP à interface virtual do túnel no Servidor A.
* "ip link set tunelab up":
  ** Ativa a interface do túnel.
* "ip route add 10.0.2.0/24 via 192.168.100.2 dev tunelab":
  ** Adiciona uma rota estática para a rede interna do Servidor B através
  do túnel,
usando o IP do túnel do Servidor B como gateway.
```

■ **Exemplo 4.189** Configuração o túnel IPIP (IP over IP) no Servidor B

```
Comandos:

ip tunnel add tunelba mode ipip remote 203.0.113.1 local 198.51.100.1 ttl 64
ip addr add 192.168.100.2/30 dev tunelba
ip link set tunelba up
ip route add 10.0.1.0/24 via 192.168.100.1 dev tunelba

Explicação das linhas de comandos:
A configuração é espelhada ao Servidor A, definindo os endereços remotos
e locais de forma inversa e a rota para a rede interna do Servidor A. A
interface "tunelba" será criada e configurada no Servidor B. Após a
configuração em ambos os lados, hosts nas redes "10.0.1.0/24" e
"10.0.2.0/24" deverão se comunicar.

Obs.: Deverá estar habilitado o roteamento em ambos os servidores
(sysctl -w net.ipv4.ip_forward=1).
```

■ **Exemplo 4.190** Comandos para verificação e remoção do túnel, mesmos comandos para os modos IPIP e GRE)

```
# Verificação
ip link show tunelab
ip -d tunnel show tunelab

# Remoção
ip link set tunelab down
ip tunnel del tunelab
```

Procedimento 4.17.2

Será realizada a configuração de um túnel do modo GRE (Generic Routing Encapsulation) entre os servidores A e B que possibilitará a comunicação entre as redes internas.

■ **Exemplo 4.191** Configuração o túnel GRE no Servidor A

```
Comandos:

ip tunnel add tunelab mode gre remote 198.51.100.1 \
   local  203.0.113.1 ttl 64 key 1
ip addr add 192.168.100.1/30 dev tunelab
ip link set tunelab up
ip route add 10.0.2.0/24 via 192.168.100.2 dev tunelab


Explicação das linhas de comando:
* "ip tunnel add tunelab mode gre":
  ** Cria uma nova interface de túnel GRE chamada ‘tunelab‘.
* ‘key 1‘:
  ** Adiciona uma "key" (chave) ao túnel. Embora não seja uma chave de
  segurança, ela ajuda a identificar o túnel e a evitar colisões, sendo
```

```
  importante para ambientes com múltiplos túneis. Ambos os lados devem
  ter a mesma "key".
* Os outros parâmetros:
  ** (‘remote‘, ‘local‘, ‘ttl‘, ‘ip addr add‘, ‘ip link set up‘,
  ‘ip route add‘) funcionam de forma idêntica ao modo IPIP.
```

■ **Exemplo 4.192** Configuração o túnel GRE no Servidor B

```
Comandos:

ip tunnel add tunelba mode gre remote 203.0.113.1 local 198.51.100.1 \
  ttl 64 key 1
ip addr add 192.168.100.2/30 dev tunelba
ip link set tunelba up
ip route add 10.0.1.0/24 via 192.168.100.1 dev tunelba


Explicação das linhas de comandos:
* Configuração espelhada para o Servidor B, garantindo que a "key" seja a
mesma.
```

■ **Exemplo 4.193** Comandos para verificação e remoção do túnel, mesmos comandos para os modos IPIP e GRE)

```
# Verificação
ip link show tunelba
ip -d tunnel show tunelba

# Remoção
ip link set tunelba down
ip tunnel del tunelba
```

5. Iptables

[1]

Neste capítulo serão apresentadas as funcionalidades do utilitário **iptables**. Este utilitário tem a função de manipular o sistema netfilter que é o firewall de pacotes do Linux. Algumas das funcionalidades mais comuns do Iptables são a construção de firewalls de rede, uso de NAT e PAT, recursos para implementação de proxies transparentes, manipulação de pacotes (tabela mangle) e edição de cabeçalhos. A base da filtragem de pacotes no Linux está em dois módulos. O módulo Netfilter possui funcionalidades de rede internas ao kernel para permitir a filtragem e manipulação de pacotes. O módulo Iptables é a ferramenta em nível de usuário para acesso ao Netfilter.

5.1 Histórico

- Primeira geração: ipfw (BSD)
- Segunda geração: ipfwadm (Linux 2.0)
- Terceira geração: ipchains (Linux 2.2)
- Quarta geração: iptables (Linux 2.4, 2.6, 3.x. 4.x)
- Quinta geração: nftables (Linux 3.13 em diante)

5.2 Componentes

- Chains
- Tabelas
- Políticas
- Ações

5.2.1 Chains

- INPUT: pacotes com destino ao host
- OUTPUT: pacotes originados do host
- FORWARD: pacotes passando pelo host
- POSTROUTING: pacotes que já passaram pelo processo de roteamento
- PREROUTING: pacotes que ainda não passaram pelo processo de roteamento

■ **Exemplo 5.1** *iptables –A INPUT –i eth0 –p tcp —dport 80 –j ACCEPT*

■ **Exemplo 5.2** *iptables –A INPUT –i eth1 –p tcp –s 192.168.10.15/32 --dport 5001 –j REJECT*

■ **Exemplo 5.3** *iptables –A OUTPUT –p udp --dport 53 –j DROP*

■ **Exemplo 5.4** *iptables –A FORWARD –i eth0 –s 200.18.79.0/24 –d 189.78.10.45/32 --dport 3389 –j ACCEPT*

■ **Exemplo 5.5** *iptables -t nat -A POSTROUTING -o eth0 -j SNAT -s 192.168.200.0/24 --to-source 200.67.10.5*

[1] https://www.youtube.com/watch?v=6a-nJ3NQccs

■ **Exemplo 5.6** *iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.200.0/24 --dport 80 -j REDIRECT --to-port 3128*

5.2.2 Tabelas

- Filter: padrão, onde é realizada a filtragem de pacotes
- NAT: onde acontecem as traduções de endereçamentos e portas dos pacotes
- Mangle: onde são modificados os cabeçalhos dos pacotes

■ **Exemplo 5.7** *iptables –A INPUT –t filter –i eth0 –p tcp --dport 80 –j ACCEPT*

■ **Exemplo 5.8** *iptables -t nat -A POSTROUTING -o eth0 -j SNAT -s 192.168.200.0/24 --to-source 200.67.10.5*

■ **Exemplo 5.9** *iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.200.0/24 --dport 80 -j REDIRECT --to-port 3128*

■ **Exemplo 5.10** *iptables -A PREROUTING -t nat -p tcp -d 177.90.17.45 --dport 18001 -j DNAT --to 192.168.200.10:80*

■ **Exemplo 5.11** *iptables -t mangle -A PREROUTING -i eth1 -p udp --dport 53 -j DSCP --set-dscp-class CS2*

■ **Exemplo 5.12** *iptables -t mangle -A POSTROUTING -o eth0 -p udp --dport 53 -j DSCP --set-dscp-class CS2*

5.2.3 Políticas

- ACCEPT: todos os pacotes que não são explicitamente bloqueados, serão aceitos
- DROP: todos os pacotes que não são explicitamente liberados, serão bloqueados

■ **Exemplo 5.13** *iptables –P INPUT DROP*

■ **Exemplo 5.14** *iptables –P OUTPUT DROP*

■ **Exemplo 5.15** *iptables –P FORWARD ACCEPT*

5.2.4 Ações

- ACCEPT: se ocorrer uma combinação com a regra, o pacote é liberado
- DROP: se ocorrer uma combinação com a regra, o pacote é bloqueado sem aviso ao remetente
- REJECT: se ocorrer uma combinação com a regra, o pacote é bloqueado e o remetente receberá um aviso
  - UDP: ICMP host ou porta inalcançável
  - TCP: segmento com a flag Reset ativada
- LOG: se ocorrer uma combinação com a regra, o pacote será registrado em arquivo Normalmente, em /var/log/messages
- REDIRECT: se ocorrer uma combinação com a regra, o pacote será redirecionando para uma porta definida na regra
- MASQUERADE: se ocorrer uma combinação com a regra, o pacote terá o endereço de origem traduzido para o endereço da interface de saída

■ **Exemplo 5.16** *iptables -A INPUT -p tcp -s 127.0.0.1 -d 127.0.0.1 -j ACCEPT*

■ **Exemplo 5.17** *iptables -A FORWARD -i eth1.100 -p icmp -s 192.168.200.0/24 -d 0/0 -j DROP*

■ **Exemplo 5.18** *iptables -A FORWARD -i eth0 -p udp -s 10.10.20.0/24 -d 10.10.30.0/24 -j REJECT*

■ **Exemplo 5.19** *iptables -A FORWARD -p tcp -s 10.10.20.15/32 -d 0/0 -m limit --limit 20/min --limit-burst 5 -j LOG --log-level 7*

■ **Exemplo 5.20** *iptables -t nat -A PREROUTING -i eth1.50 -p tcp -s 192.168.200.0/24 --dport 80 -j REDIRECT --to-port 3128*

■ **Exemplo 5.21** *iptables -A POSTROUTING -t nat -o eth0 -s 192.168.200.0/24 -j MASQUERADE*

5.3 Manipulação de Regras

Parâmetros -A, -I e -D:

- "-A" insere a regra abaixo da última regra adicionada.
- "-I" posiciona a regra no topo da lista de regras atual
- "-D" remove a regra

■ **Exemplo 5.22** *iptables -A FORWARD -p tcp -s 10.10.20.15/32 -d 0/0 -j ACCEPT*

■ **Exemplo 5.23** *iptables –I FORWARD -p tcp -s 10.10.20.15/32 -d 0/0 -j ACCEPT*

■ **Exemplo 5.24** *iptables –D FORWARD -p tcp -s 10.10.20.15/32 -d 0/0 -j ACCEPT*

- Parâmetro –L –n –v : para listar as regras, sem tradução de nomes e com os contadores de Bytes e pacotes em cada regra
- Parâmetro –F: para limpar as regras das tabelas filter, mangle e nat

■ **Exemplo 5.25** *iptables –L –n –v*

■ **Exemplo 5.26** *iptables -F -t filter*

■ **Exemplo 5.27** *iptables -F -t nat*

■ **Exemplo 5.28** *iptables -F -t mangle*

5.4 Valores padrão

Todo o parâmetro não atribuído, é assumido o valor qualquer um (any) ou o valor todos Se não for definido um endereço de origem, destino, interface, protocolo ou porta, será assumido qualquer endereço de origem e destino, porta, protocolo e todas as interfaces.

■ **Exemplo 5.29** *iptables –A INPUT –j ACCEPT*

■ **Exemplo 5.30** *iptables –A FORWARD –p tcp –j DROP*

■ **Exemplo 5.31** *iptables -A FORWARD -p udp –s 0/0 —dport 161 –j ACCEPT*

5.5 Módulos

- Multiport: para colocar múltiplas portas na regra
- Limit: para limitar o número de combinações da regra
- State: para criar combinações conforme o estado do fluxo de pacotes Ativa o modo stateful, onde o estado do fluxo de pacotes é mantido pelo firewall

■ **Exemplo 5.32** *iptables -A FORWARD -p tcp -m multiport -s 10.0.0.10/32 -d 0/0 --dports 80,443,465,993,995,587,8080,8081 -j ACCEPT*

■ **Exemplo 5.33** *iptables -A FORWARD -p tcp -s 10.0.0.10/32 -d 0/0 -m limit --limit 20/min --limit-burst 5 -j LOG --log-level 7*

■ **Exemplo 5.34** *iptables -I INPUT -p udp -s 10.15.0.0/16 -d 0/0 --dport 53 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT*

- Connlimit: para limitar o número de conexões de determinada regra
- Mac: para criar regras baseadas no endereço físico (mac address)
- Iprange: para usar em faixas de IPs
- String: para usar em regras que combinam palavras nos pacotes

■ **Exemplo 5.35** *iptables -A INPUT -p tcp --syn --dport 22 -m connlimit --connlimit-above 3 -j REJECT*

■ **Exemplo 5.36** *iptables -A FORWARD -m mac --mac-source 00:0F:EA:91:04:08 -j ACCEPT*

■ **Exemplo 5.37** *iptables -A FORWARD -p tcp -m iprange --src-range 192.168.1.5-192.168.1.100 --dport 143 -j ACCEPT*

■ **Exemplo 5.38** *iptables -A FORWARD -m string --algo bm --string "facebook" –p udp –dport 53 -j DROP*

5.6 Outros Exemplos

NAT 1:1: para traduzir um endereço IP interno para um endereço IP externo, quando se tem mais de um IP público

■ **Exemplo 5.39**
```
# NAT 1:1 (10.15.160.25 -> 200.18.16.13)
iptables -A FORWARD -s 10.15.160.25/32 -j ACCEPT
iptables -A FORWARD -d 10.15.160.25/32 -j ACCEPT
iptables -A INPUT -s 10.15.160.25/32 -j ACCEPT
iptables -A POSTROUTING -t nat -s 10.15.160.25/32 -o eth0 -j SNAT --to 200.18.16.13
iptables -A PREROUTING -t nat -d 200.18.16.13 -j DNAT --to 10.15.160.25
```

Redirecionamento de portas para outro host: a ação REDIRECT permite apenas o redirecionamento de portas para o próprio firewall. Com o uso de ferramentas tais como redir ou socat junto ao iptables, é possível fazer o redirecionamento para uma porta em um host remoto

■ **Exemplo 5.40**
```
# Redirecionamento de portas e hosts
# Instalar o utilitário redir
redir --lport=8000 --caddr=192.168.200.6 --cport=80
iptables -t nat -I PREROUTING -i eth1 -p tcp -d 192.168.200.3 --dport 80 -j REDIRECT --to-port 8000
```

5.7 Exemplo de Script de Firewall Completo

Exemplo de script de firewall para o cenário da Figura 5.1. Neste cenário são apresentados dois firewalls que atendem as empresas Empresa1 e Empresa2.

5.7.1 Firewall da Empresa 1

Link para download: HTTPS://TINYURL.COM/RJCDDJEN

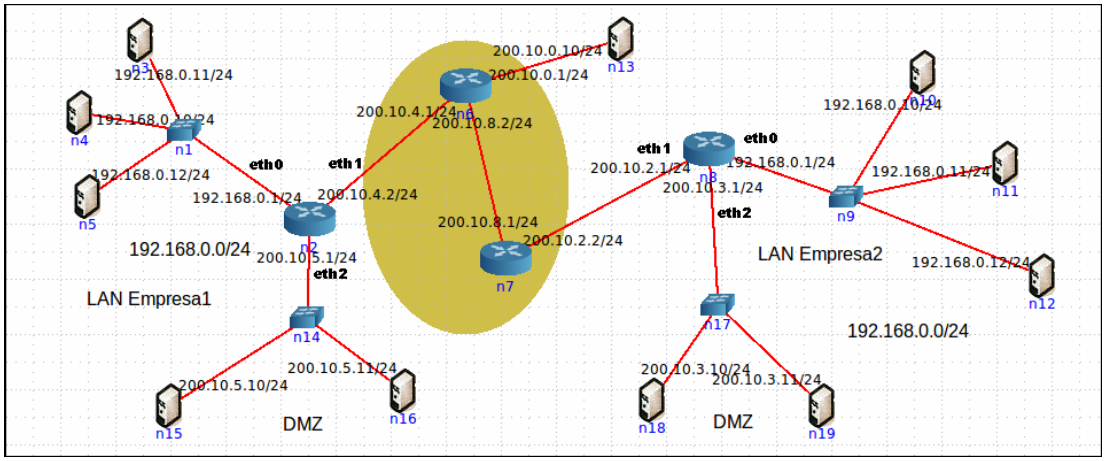

Figure 5.1: Cenário exemplo para script de firewall com Iptables.

```
#!/bin/bash
# Firewall da empresa 1
# NAT - rede interna para a interface externa
# eth1 - interface externa (WAN)
# eth0 - interface interna (LAN)
# eth2 - interface DMZ
# Ativa o roteamento
echo 1 > /proc/sys/net/ipv4/ip_forward
#### Ativa Modulos
        /sbin/modprobe ip_conntrack
        /sbin/modprobe ip_conntrack_ftp
        /sbin/modprobe ip_nat_ftp
        /sbin/modprobe ipt_LOG
#### Limpa tabelas e configura defaults
        iptables -F -t filter
        iptables -F -t nat
        iptables -F -t mangle
        ## Delete chains nao defaults
        iptables -X
        iptables -X -t nat
        iptables -X -t mangle
        # Política DROP, chains INPUT e FORWARD
        iptables -P INPUT  DROP -t filter
        iptables -P OUTPUT ACCEPT -t filter
        iptables -P FORWARD DROP -t filter
  # para manter as conexoes existentes com o proprio firewall (entrada)
        iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
        # para manter as conexoes existentes com o proprio firewall (saida)
        iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
  # para manter as conexoes passando pelo firewall
        iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT


# Libera acesso ao ICMP (Ping) nas interfaces LAN, DMZ e WAN
iptables -A INPUT -i eth0 -p icmp -j ACCEPT
iptables -A INPUT -i eth1 -p icmp -j ACCEPT
iptables -A INPUT -i eth2 -p icmp -j ACCEPT
# Libera SSH na porta 22
```

```
iptables -A INPUT -i eth1 -p tcp  -s 0/0 --dport 22 -j ACCEPT
iptables -A INPUT -i eth0 -p tcp  -s 0/0 --dport 22 -j ACCEPT
iptables -A INPUT -i eth2 -p tcp  -s 0/0 --dport 22 -j ACCEPT
# Libera a resolucao de DNS no servidor local  (LAN e DMZ)
iptables -A INPUT -i eth0 -p udp -s 192.168.0.0/24 -d 0/0 --dport 53 -m state
--state NEW,ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -i eth2 -p udp -s 200.10.5.0/24 -d 0/0 --dport 53 -m state
--state NEW,ESTABLISHED,RELATED -j ACCEPT
# Mantem o estado das conexoes da interface de loopback
iptables -A INPUT  -s 127.0.0.1  -m state --state RELATED,ESTABLISHED
-j ACCEPT
iptables -A OUTPUT  -s 127.0.0.1  -m state --state RELATED,ESTABLISHED
-j ACCEPT
# Libera todos os acesso originados de localhost para localhost
iptables -A INPUT -s 127.0.0.1 -d 127.0.0.1  -j ACCEPT
# Ativa mascaramento (rede interna para o IP da WAN) (IP Fixo)
iptables -t nat -A POSTROUTING -o eth1 -j SNAT -s 192.168.0.0/24
--to-source 200.10.4.2

### Liberacoes (rede interna (LAN) para fora)
# Servicos comuns
iptables -A FORWARD -i eth0 -p tcp -m multiport -s 192.168.0.0/24
-d 0/0 --dports 22,80,443,5001 -j ACCEPT
# Host 192.168.0.12 bloqueado ICMP para fora
iptables -A FORWARD -i eth0 -p icmp -s 192.168.0.12/32 -d 0/0 -j DROP
# ICMP
iptables -A FORWARD -i eth0 -p icmp -s 192.168.0.0/24 -d 0/0 -j ACCEPT
# DNS, NTP
iptables -A FORWARD -i eth0 -p udp -s 192.168.0.0/24 -d 0/0
-m multiport --dports 53,143
-m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
# Redireciona portas (porta publica: 5001, IP interno: 192.168.0.10,
# Porta interna: 5001)
iptables -A FORWARD -i eth1 -d 192.168.0.10/32 -j ACCEPT
iptables -A PREROUTING -t nat -p tcp -d 200.10.4.2 --dport 5001 -j DNAT
--to 192.168.0.10:5001


### Liberacoes (rede DMZ para fora)
# Servicos comuns
iptables -A FORWARD -i eth2 -p tcp -m multiport -s 200.10.5.0/24 -d 0/0
--dports 22,80,443,5001 -j ACCEPT
# ICMP
iptables -A FORWARD -i eth2 -p icmp -s 200.10.5.0/24 -d 0/0 -j ACCEPT
# DNS, NTP
iptables -A FORWARD -i eth0 -p udp -s 200.10.5.0/24 -d 0/0 -m multiport
--dports 53,143  -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT



### Liberacoes (rede externa para a DMZ)
# Host externo 200.10.0.10, bloqueado o acesso a porta 22 no servidor
#200.10.5.10
iptables -A FORWARD -i eth1 -p tcp -s 200.10.0.10/32 -d 200.10.5.10/32
--dport 22 -j DROP
# Servicos comuns, somente para o servidor 200.10.5.10
```

```
iptables -A FORWARD -i eth1 -p tcp -m multiport -s 0/0 -d 200.10.5.10/32
--dports 22,80,443,5001 -j ACCEPT
# SSH e HTTP, somente para o servidor 200.10.5.11
iptables -A FORWARD -i eth1 -p tcp -m multiport -s 0/0 -d 200.10.5.11/32
--dports 22,80 -j ACCEPT
# ICMP
iptables -A FORWARD -i eth1 -p icmp -s 200.10.5.0/24 -d 0/0 -j ACCEPT
# Host externo 200.10.0.10, acesso a porta 5001 no servidor 200.10.5.11
iptables -A FORWARD -i eth1 -p tcp -s 200.10.0.10/32 -d 200.10.5.11/32
--dport 5001 -j ACCEPT
```

5.7.2 Firewall da Empresa 2

Link para download: HTTPS://TINYURL.COM/MT4UBS5B

```
#!/bin/bash
# Firewall da empresa 2
# NAT - rede interna para a interface externa
# eth1 - interface externa (WAN)
# eth0 - interface interna (LAN)
# eth2 - interface DMZ
# Ativa o roteamento
echo 1 > /proc/sys/net/ipv4/ip_forward
#### Ativa Modulos
        /sbin/modprobe ip_conntrack
        /sbin/modprobe ip_conntrack_ftp
        /sbin/modprobe ip_nat_ftp
        /sbin/modprobe ipt_LOG
#### Limpa tabelas e configura defaults
        iptables -F -t filter
        iptables -F -t nat
        iptables -F -t mangle
        ## Delete chains nao defaults
        iptables -X
        iptables -X -t nat
        iptables -X -t mangle
        # Política DROP, chains INPUT e FORWARD
        iptables -P INPUT  DROP -t filter
        iptables -P OUTPUT ACCEPT -t filter
        iptables -P FORWARD DROP -t filter
  # para manter as conexoes existentes com o proprio firewall (entrada)
        iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
        # para manter as conexoes existentes com o proprio firewall (saida)
        iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
  # para manter as conexoes passando pelo firewall
        iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
# Libera acesso ao ICMP (Ping) nas interfaces LAN, DMZ e WAN
iptables -A INPUT -i eth0 -p icmp -j ACCEPT
iptables -A INPUT -i eth1 -p icmp -j ACCEPT
```

```
iptables -A INPUT -i eth2 -p icmp -j ACCEPT
# Libera SSH na porta 22
iptables -A INPUT -i eth1 -p tcp  -s 0/0 --dport 22 -j ACCEPT
iptables -A INPUT -i eth0 -p tcp  -s 0/0 --dport 22 -j ACCEPT
iptables -A INPUT -i eth2 -p tcp  -s 0/0 --dport 22 -j ACCEPT
# Libera a resolucao de DNS no servidor local  (LAN e DMZ)
iptables -A INPUT -i eth0 -p udp -s 192.168.0.0/24 -d 0/0 --dport 53
-m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -i eth2 -p udp -s 200.10.3.0/24 -d 0/0 --dport 53
-m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
# Mantem o estado das conexoes da interface de loopback
iptables -A INPUT  -s 127.0.0.1  -m state --state RELATED,ESTABLISHED
-j ACCEPT
iptables -A OUTPUT  -s 127.0.0.1  -m state --state RELATED,ESTABLISHED
-j ACCEPT
# Libera todos os acesso originados de localhost para localhost
iptables -A INPUT  -s 127.0.0.1 -d 127.0.0.1  -j ACCEPT
# Ativa mascaramento (rede interna para o IP da WAN) (IP Fixo)
iptables -t nat -A POSTROUTING -o eth1 -j SNAT -s 192.168.0.0/24
--to-source 200.10.2.1

### Liberacoes (rede interna (LAN) para fora)
# Servicos comuns
iptables -A FORWARD -i eth0 -p tcp -m multiport -s 192.168.0.0/24 -d 0/0
--dports 22,80,443,5001 -j ACCEPT
# Host 192.168.0.12 bloqueado ICMP para fora
iptables -A FORWARD -i eth0 -p icmp -s 192.168.0.12/32 -d 0/0 -j DROP
# ICMP
iptables -A FORWARD -i eth0 -p icmp -s 192.168.0.0/24 -d 0/0 -j ACCEPT
# DNS, NTP
iptables -A FORWARD -i eth0 -p udp -s 192.168.0.0/24 -d 0/0 -m multiport
--dports 53,143
-m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
# Redireciona portas (porta publica: 5001, IP interno: 192.168.0.10,
#Porta interna: 5001)
iptables -A FORWARD -i eth1 -d 192.168.0.10/32 -j ACCEPT
iptables -A PREROUTING -t nat -p tcp -d 200.10.2.1 --dport 5001 -j DNAT
--to 192.168.0.10:5001



### Liberacoes (rede DMZ para fora)
# Servicos comuns
iptables -A FORWARD -i eth2 -p tcp -m multiport -s 200.10.3.0/24 -d 0/0
--dports 22,80,443,5001 -j ACCEPT
# ICMP
iptables -A FORWARD -i eth2 -p icmp -s 200.10.3.0/24 -d 0/0 -j ACCEPT
# DNS, NTP
iptables -A FORWARD -i eth0 -p udp -s 200.10.3.0/24 -d 0/0 -m multiport
--dports 53,143  -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

### Liberacoes (rede externa para a DMZ)
# Host externo 200.10.0.10, bloqueado o acesso a porta 22 no servidor
# 200.10.3.10
iptables -A FORWARD -i eth1 -p tcp -s 200.10.0.10/32 -d 200.10.3.10/32
```

```
--dport 22 -j DROP
# Servicos comuns, somente para o servidor 200.10.3.10
iptables -A FORWARD -i eth1 -p tcp -m multiport -s 0/0 -d 200.10.3.10/32
--dports 22,80,443,5001 -j ACCEPT
# SSH e HTTP, somente para o servidor 200.10.3.11
iptables -A FORWARD -i eth1 -p tcp -m multiport -s 0/0 -d 200.10.3.11/32
--dports 22,80 -j ACCEPT
# ICMP
iptables -A FORWARD -i eth1 -p icmp -s 200.10.3.0/24 -d 0/0 -j ACCEPT
# Host externo 200.10.0.10, acesso a porta 5001 no servidor 200.10.3.11
iptables -A FORWARD -i eth1 -p tcp -s 200.10.0.10/32 -d 200.10.3.11/32
--dport 5001 -j ACCEPT
```

5.7.3 Firewall da Empresa 1 - versão IPv6

Link para download: HTTPS://TINYURL.COM/2SR4ZX8E

```
#!/bin/bash
# Firewall da empresa 1
# NAT - rede interna para a interface externa
# eth1 - interface externa (WAN)
# eth0 - interface interna (LAN)
# eth2 - interface DMZ
# Firewall simples para IPv6

# Ativar roteamento
echo 1 > /proc/sys/net/ipv6/conf/all/forwarding

# Limpa regras
ip6tables -F INPUT
ip6tables -F OUTPUT
ip6tables -F FORWARD
ip6tables  -F -t mangle
ip6tables -F

# Politica padrao DROP
ip6tables -P INPUT DROP
ip6tables -P OUTPUT ACCEPT
#ip6tables -P FORWARD DROP

# Estados das conexoes
ip6tables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
ip6tables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
ip6tables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Libera acesso localhost
ip6tables -A INPUT -i lo -j ACCEPT

# Libera enderecos Link-Local
```

```
ip6tables -A INPUT -s fe80::/10 -j ACCEPT

# Libera multicast
ip6tables -A INPUT -d ff00::/8 -j ACCEPT

# Libera ICMP
ip6tables -A INPUT -p icmpv6 -s 2001:6::/48 -j ACCEPT
ip6tables -A INPUT -p icmpv6 -s 2001:4::/64 -j ACCEPT

#               ============================
#               ### Libera HTTP e HTTPs ###
#               ============================

ip6tables -A FORWARD -i eth0 -p tcp  -s ::/0 -m multiport
--dport 80,443,8080,5001 -j ACCEPT


# ICMP
ip6tables -A INPUT -i eth0 -p icmpv6  -s ::/0  -j ACCEPT

# SSH
ip6tables -A FORWARD -i eth0 -p tcp -d 2001:5:10:102::/64
--dport 50000 -j ACCEPT

# Libera acesso de qualquer origem para a porta TCP 5001 do firewall
ip6tables -A INPUT -i eth0 -p tcp -s ::/0 --dport 5001 -j ACCEPT

# Libera acesso para um host realizar acesso ao protocolo RDP
ip6tables -A FORWARD -i eth0 -p tcp -s 2001:2::161:0:8/128 -m multiport
--dport 3389 -j ACCEPT


# Logs
#ip6tables -A INPUT -j LOG --log-prefix "Firewall:input:DROP:"
-m limit --limit 20/min --limit-burst 5 --log-level 7
```

6. Docker Networking

Neste capítulo serão apresentados os comandos, procedimentos e configurações de redes para uso com a plataforma Docker (https://www.docker.com/. O Docker é uma plataforma de software de código-fonte aberto que permite aos desenvolvedores criar, executar, atualizar, implantar e gerenciar aplicativos em contêineres. A comunicação entre os contêineres e o restante da rede é realizada por meio de comandos e interfaces de rede específicas.

- **Bridge Network (Rede de Ponte)**: A bridge network é a rede padrão do Docker. Ela permite que os contêineres se comuniquem entre si e com o host em uma rede privada isolada. Para cada contêiner será atribuído um endereço IP na sub-rede específica da bridge. Os contêineres podem se conectar a essa rede usando o parâmetro "--network" bridge ao criar um novo contêiner.
- **Host Network (Rede de Hospedeiro)**: Com a opção de rede de hospedeiro, o contêiner compartilha a pilha de rede do host, o que significa que ele pode acessar a rede do host diretamente, sem a necessidade de mapeamento de portas ou NAT. Isso pode ser útil para aplicativos que precisam de um alto desempenho de rede, mas pode comprometer o isolamento do contêiner.
- **Overlay Network (Rede de Sobreposição)**: A rede de sobreposição é usada para conectar contêineres distribuídos em diferentes hosts em um cluster chamado de Docker Swarm. Será criada uma rede virtual sobre a infraestrutura física subjacente, permitindo a comunicação entre contêineres em diferentes hosts. Essa configuração é ideal para aplicativos distribuídos e escaláveis.
- **Macvlan Network (Rede Macvlan)**: A rede Macvlan permite que os contêineres tenham endereços MAC e IP próprios, tornando-os visíveis diretamente na rede física. Isso é útil quando os contêineres precisam aparecer como dispositivos físicos na rede, como para integração com sistemas legados ou quando é necessário acesso direto à rede física.
- **Custom Bridge Network (Rede de Ponte Personalizada)**: Podem ser criadas redes de ponte personalizadas, configurando sub-redes específicas, endereços IP fixos, configurações de DNS e outros parâmetros. Isso permite uma maior flexibilidade na configuração da rede para atender às necessidades específicas de algum aplicativo.
- **IPvlan**: Provê o isolamento da rede do contêiner, onde haverá um endereço MAC próprio e um endereçamento diferente do hospedeiro. Funciona na camada 2 e possibilita a comunicação direta do contêiner com a rede física, ignorando a comunicação de rede do hospedeiro. Funciona de forma similar ao modo bridged em soluções de virtualização.

A rede do tipo Bridge funciona tal como o tipo de interface NAT em soluções de virtualização. Neste tipo de rede os contêineres se comunicam por uma rede interna isolada e fazem acesso ao resto da rede pelo endereço do hospedeiro do Docker. A rede interna é provida por um servidor de DHCP do próprio Docker que faz a distribuição dos endereços para os contêineres. As configurações dos servidores de DNS são herdadas do sistema hospedeiro, por exemplo a partir do arquivo /etc/resolv.conf em sistemas operacionais Linux.

Procedimento 6.0.1

Comandos básicos para administração de contêineres.

■ **Exemplo 6.1** Listar contêineres

```
docker ps
```

■ **Exemplo 6.2** Para listar as imagens criadas (tanto as que foram baixadas, quanto as "comitadas"):

```
docker images
```

■ **Exemplo 6.3** Para criar um contêiner de nome "ServBD1" com a imagem do Ubuntu 14.04.2 (caso não tenha a imagem baixada, é feito o download do próprio repositório docker)

```
docker run -it --name ServBD1 ubuntu:14.04.2 /bin/bash
```

■ **Exemplo 6.4** Para criar um contêiner e abrir uma porta para o host hospedeiro, porta 2222 (porta externa) para a porta 22 (interna), porta 55432 (externa) para a porta 5432 (interna)

```
docker run -it -p  2222:22 -p 55432:5432 ubuntu:14.04.2 /bin/bash
```

■ **Exemplo 6.5** Para sair do contêiner e deixá-lo em execução (caso contrário, o contêiner será destruído e a instância finalizada)

```
Combinação de teclas:

CTRL+p + CTRL+q
```

■ **Exemplo 6.6** Para remover uma imagem com o ID 7e6df4470869 (deletar permanentemente)

```
docker rmi -f 7e6df4470869
```

■ **Exemplo 6.7** Para remover um contêiner (destruir)

```
docker rm -f 35ddd53bfdfd
```

■ **Exemplo 6.8** Acessar a console do contêiner com ID 31c4fc57d564

```
docker container attach 31c4fc57d564
```

■ **Exemplo 6.9** Acessar o interpretador de comandos bash no contêiner de ID 31c4fc57d564

```
docker exec -it 31c4fc57d564 bash
```

■ **Exemplo 6.10** Listar informações de um contêiner com o ID 31c4fc57d564, com detalhes sobre as configurações de rede

```
docker inspect 31c4fc57d564
```

■ **Exemplo 6.11** Listar as redes disponíveis

```
docker network ls
```

■ **Exemplo 6.12** Listar informações sobre uma rede de nome "lan-net"

```
docker network inspect lan-net
```

■ **Exemplo 6.13** Cria uma rede com nome de "interno-net" do tipo bridge

```
docker network create --driver bridge interno-net
```

■ **Exemplo 6.14** Reiniciar contêiner de nome bd-app

```
docker restart bd-app
```

■ **Exemplo 6.15** Rodar sem rede (rede do tipo "none")

```
docker run -it --rm --network none busybox:latest
```

■ **Exemplo 6.16** Remover rede de um contêiner, rede interno-net do contêiner "container2"

```
docker network disconnect interno-net container2
```

■ **Exemplo 6.17** Para remover uma rede de nome "interno-net"

```
docker network rm interno-net
```

■ **Exemplo 6.18** Abrir uma porta em um contêiner em execução

```
# Abre a porta TCP 9300 para acesso externo do contêiner com IP 172.17.0.3

iptables -t nat -A DOCKER -p tcp --dport 9300 -j DNAT \
--to-destination 172.17.0.3:9300

# Abre a porta TCP 9301 para acesso externo do contêiner com
# IP 172.18.0.3 e porta TCP 5000

iptables -t nat -A DOCKER -p tcp --dport 9300 -j DNAT \
--to-destination 172.18.0.3:5000
```

■ **Exemplo 6.19** Utiliza o mesmo IP do hospedeiro com o tipo de rede "host"

```
docker run -d --name nginx --network host nginx:latest
```

■ **Exemplo 6.20** Cria a rede "BD-interno" entre dois contêineres e faz a associação da rede. Esta rede será utilizada para comunicação exclusiva entre os dois contêineres ServBD1 e ServDB2.

```
docker network create --driver bridge BD-interno
docker network connect BD-interno ServBD1
docker network connect BD-interno ServBD2
```

■ **Exemplo 6.21** Cria uma rede do tipo IPVLAN de nome lan-net com o prefixo de rede 192.168.254.0/24.

```
 docker network create -d ipvlan --subnet=192.168.254.0/24 \
     --gateway=192.168.254.254 \
     -o ipvlan_mode=l2 \
     -o parent=ens33 lan-net
Exemplos de aplicação:

docker run --net=lan-net -it --rm alpine /bin/sh
docker run --net=lan_net --ip=192.168.254.61 -it --rm alpine /bin/sh
docker run --net=lan_net -p 7500:7500 --ip=192.168.254.62 -it \
--rm alpine /bin/sh
```

Procedimento 6.0.2 Instalação do Docker no Debian 11

Passos para a instalação da versão estável mais recente do Docker na distribuição Debian 11.

```
 apt remove docker docker-engine docker.io containerd runc

 apt update

 apt install ca-certificates curl gnupg lsb-release

 curl -fsSL https://download.docker.com/linux/debian/gpg | \
 sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

 echo "deb [arch=\$(dpkg --print-architecture) \
 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] \
 https://download.docker.com/linux/debian $(lsb\_release -cs) stable" | \
 sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

 apt update

 apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin \
 docker-compose-plugin

 docker run hello-world
```

7. Anexo

Neste capítulo serão apresentados procedimentos e ferramentas auxiliares para a administração de sistemas operacionais Linux em rede.

7.1 systemctl

O gerenciamento da inicialização e dos serviços em Linux é feito pelo systemd Padrão nas distribuições atuais. A forma clássica de gerenciamento de serviços usava os scripts init.d e com a ferramenta service. Exemplos:

- /etc/init.d/apache2 restart
- services apache2 restart

Basicamente, são scripts que gerenciam a execução dos processos que ficam em segundo plano (daemon)

As opções padrão para gerenciar os serviços são:

- **start** – para iniciar um serviço
- **stop** – para interromper um serviço
- **restart** – para reiniciar um serviço
- **reload** – para recarregar as configurações de um serviço
- **status** – para verificar o estado da execução de um processo

A opção de reload não termina o processo em execução, o objetivo é fazer um recarregamento para atualizar configurações. A opção restart encerra o processo e inicia outro processo.

O comando systemctl é usado para gerenciar os serviços em sistemas com systemd. Exemplos de uso para o serviço do Apache:

Para iniciar o serviço

■ **Exemplo 7.1** *systemctl start apache2*

Para interromper o serviço

■ **Exemplo 7.2** *systemctl stop apache2*

Para reiniciar o serviço

■ **Exemplo 7.3** *systemctl restart apache2*

Para recarregar as configurações do serviço

■ **Exemplo 7.4** *systemctl reload apache2*

Para verificar o estado da execução do processo

■ **Exemplo 7.5** *systemctl status apache2*

Não haverá saída na tela ao executar a inicialização de um serviço que aconteça com sucesso. Para verificar se o serviço está executando, utilizar a opção status, buscar os processos no sistema ou verificar nos logs.

A forma de gerenciamento com o systemctl permite outras opções que facilitam a administração dos serviços. Exemplos de uso:

Para habilitar o serviço do Apache2 na inicialização do sistema

■ **Exemplo 7.6** *systemctl enable apache2*

Para desabilitar o serviço do Apache na inicialização do sistema

■ **Exemplo 7.7** *systemctl disable apache2*

Para listar todas os scripts e dispositivos gerenciados pelo systemd

■ **Exemplo 7.8** *systemctl list-units --all*

Para listar os scripts dos serviços

■ **Exemplo 7.9** *systemctl*

O formato dos scripts de serviços seguem um padrão, possuem a extensão ".service" e ficam armazenados em /lib/systemd/system. Outros usos que podem ser feitos com o comando systemctl.
Ao adicionar ou alterar um script de serviços deverá ser feito um reload

■ **Exemplo 7.10** *systemctl daemon-reload*

Para habilitar o modo gráfico na inicialização do sistema

■ **Exemplo 7.11** *systemctl set-default graphical.target*

Para habilitar o modo texto multiusuário na inicialização do sistema

■ **Exemplo 7.12** *systemctl set-default runlevel3.target*

Para reiniciar o sistema

■ **Exemplo 7.13** *systemctl reboot*

O nível de execução (runlevel) determina qual o modo de interação com o sistema. O nível 5 é modo gráfico.

7.2 journalctl

O systemd possui uma forma particular (journald) para verificação dos logs de mensagens das execuções dos scripts de serviços. A ferramenta journalctl é usada para manipular os logs. Exemplos de uso:
Mostra os logs em ordem cronológica

■ **Exemplo 7.14** *journalctl*

Mostra os logs em ordem cronológica inversa desde o último boot do sistema

■ **Exemplo 7.15** *journalctl –b –r*

Lista os boots realizados pelo sistema

■ **Exemplo 7.16** *journalctl --list-boots*

Mostra os logs a partir de 10/01/2023 às 13:45:00 até agora

■ **Exemplo 7.17** *journalctl --since "2023-01-10 13:45:00"*

Mostra os logs desde ontem até agora, com explicações das mensagens (quando disponível)

■ **Exemplo 7.18** *journalctl –x --since yesterday*

Mostra os logs desde às 7 horas e apenas até uma hora atrás

■ **Exemplo 7.19** *journalctl --since 07:00 --until "1 hour ago"*

Mostra todos os logs relacionados ao serviço do Apache

■ **Exemplo 7.20** *journalctl -u apache2.service*

Mostra os últimos 20 logs

■ **Exemplo 7.21** *journalctl -n 20*

Mostra os logs mais recentes de forma contínua

■ **Exemplo 7.22** *journalctl -f*

7.3 Vmware Player: Configurações de Interfaces de Rede

O virtualizador Vmware Player possibilita tipos diferentes de operação paras as interfaces de rede das máquinas virtuais. Os tipos principais são:

- **Bridged**: a máquina virtual será um host na mesma rede do hospedeiro, tal como estivesse conectada na mesma rede, seria similar a ligar a VM no mesmo switch do hospedeiro. Desta forma, os endereços usados na VM ficarão visíveis para os demais hosts da rede física usada pelo hospedeiro.
- **NAT (*Network Address Translation*)**: a máquina virtual compartilha o endereço IP do host físico. Ela usa o endereço IP do host para se comunicar com a rede externa, não ficando alcançável por hosts externos a rede usada pelo modo NAT. Por exemplo, se duas VMs utilizarem interfaces em modo NAT, elas podem se enxergar internamente por meio desta rede. É o padrão quando é criada uma máquina virtual.
- **Host-Only (Somente Hospedeiro)**: a máquina virtual pode se comunicar apenas com o host físico e outras máquinas virtuais hospedadas no mesmo host. Não há conexão com a rede externa.
- **LAN Segment**: ao criar um LAN Segment as máquinas que participam deste segmento ficam isoladas. Por exemplo, ao criar dois segmentos, LAN1 e LAN2, as máquinas que estiverem com interfaces de rede na LAN1 não enxergam as máquinas virtuais com interfaces na LAN2. Não há conexão com a rede externa.

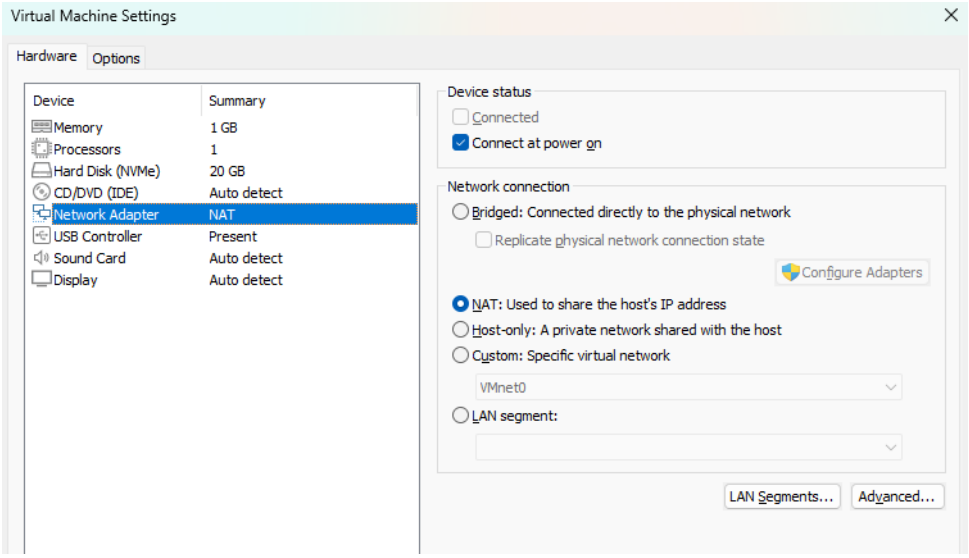

Figure 7.1: Vmware Player: tipos de interfaces e conexões de rede.

Quando o tipo de interface de rede form **Bridged** deve-se escolher qual interface do hospdeiro será usada. Nem todos os drivers de interfaces de rede sem fios aceitam o modo de bridge. Outra

observação é verificar se a interface correta está sendo usada, por exemplo, em hospedeiros com o Vmware Player e o Virtualbox instalados é comum a interface virtual do Virtualbox ser selecionada como padrão para realizar o modo bridge. Na Figura 7.2 um exemplo de opções de interfaces para seleção em uso de modo bridge.

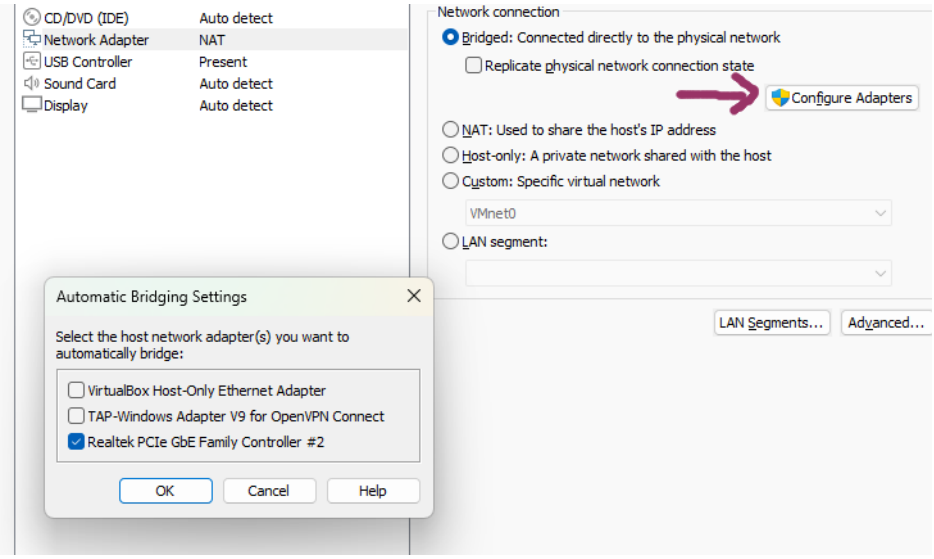

Figure 7.2: Vmware Player: escolha da interface do hospedeiro para ser usada no modo bridged.

No modo **Advanced**, para cada interface é possível configurar parâmetros de desempenho tais como largura de banda, perdas, atraso, alterar o endereço MAC (o padrão é ser gerado de forma aleatória com o prefixo *00:0C:29*:XX:XX:XX reservado para o Vmware). Usar a informação do MAC é importante para identificar qual é cada interface na máquina virtualizada (nem sempre estará na ordem que aparece no virtualizador). Na Figura 7.3 são mostrados os parâmetros disponíveis no modo avançado de configuração da interface de rede selecionada como Host-Only.

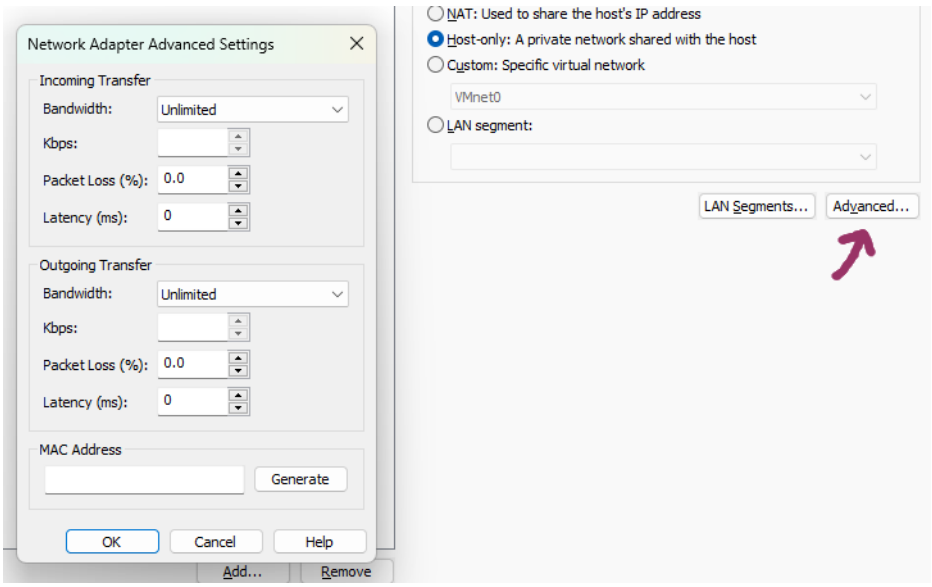

Figure 7.3: Vmware Player: parâmetros da interface de rede em modo avançado.

7.3.1 Vídeos Relacionados

- Configuração de bridge no Vmware Player [1]
- Vmware Player - configuração para firewall/roteador [2]
- Procedimentos para Criação de VMs usando os virtualizadores Vmware Player e Virtualbox [3]
- CentOS 8: instalação em VM [4]
- Debian 11: instalação em VM [5]

7.4 VirtualBox: Configurações de Interfaces de Rede

O virtualizador VirtualBox possibilita tipos diferentes de operação para as interfaces de rede das máquinas virtuais. Os tipos principais são:

- **NAT (Network Address Translation**: Permite que a máquina virtual acesse a rede externa através do endereço IP do host físico. Essa é a opção padrão.
- **Bridged Adapter**: Faz com que a máquina virtual apareça como um dispositivo independente na rede física. Seria similar a ligar a VM no mesmo switch do hospedeiro.
- **Internal Network (Rede Interna)**: Conecta a máquina virtual apenas a outras máquinas virtuais no VirtualBox, criando uma rede isolada. Podem ser criadas várias redes internas, onde as VMs de uma rede interna não enxergam VMs em outra rede interna.
- **Host-only Adapter (Adaptador Somente Hospedeiro)**: Cria uma rede privada entre a máquina virtual e o host físico, permitindo a comunicação apenas entre eles e outras VMs que utilizem uma interface host-only. Importante observar que todas as VMs que estejam com uma interface host-only participam de uma única rede.

Existem outras opções de configurações para as interfaces de rede, dependendo da versão e do sistema operacional onde está sendo executado o VirtualBox. Na Figura 7.4 são apresentadas as opções de interface de rede na versão 7.0.6 do VirtualBox sendo executado em um sistema operacional Microsoft Windows 11.

Quando for utilizada a interface em modo Bridged deve-se selecionar qual interface de rede do hospedeiro será usada para isto. Esta etapa é importante e nem toda a interface de rede possibilita o uso em modo bridge, especialmente, interfaces de rede sem fios no hospedeiro. Na

[1] https://www.youtube.com/watch?v=iXuLjxEllLk
[2] https://www.youtube.com/watch?v=3Ra99xEIbPs
[3] https://youtu.be/TnCmMTok7Go?si=mmy6Smrx2kOx65BK
[4] https://www.youtube.com/watch?v=_mYl4qOIWWo
[5] https://www.youtube.com/watch?v=iPsIP1HjUlQ

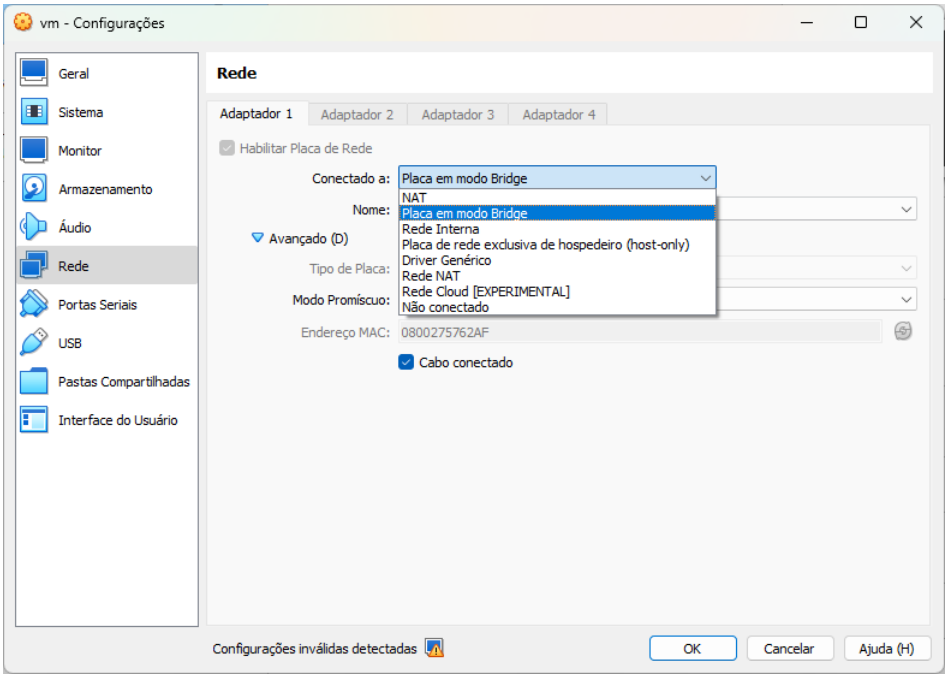

Figure 7.4: VirtualBox: opções de configuração para as interfaces de rede.

Figura 7.5 são listadas as opções de rede do hospedeiro para escolha de qual será usada em modo bridged.

No modo Avançado (Advanced) é possível escolher o driver da interface de rede que será disponibilizada para a máquina virtual, o tipo de comportamento da placa em relação a tráfego do hospedeiro e de outras VMs (modo promíscuo) e o endereço físico a ser configurado na interface da VM. Na Figura 7.6 mostra as opções de funcionamento do modo de operação da interface para uso ou não de modo promíscuo. Em modo promíscuo ativado, torna-se possível capturar tráfego de rede de todas as VMs que estejam em modo bridged ou NAT. Esta funcionalidade possibilita realizar uso de análise de tráfego com o Wireshark ou disponibilizar serviços de rede que fazem uso de broadcast/multicast tal como DHCP entre as VMs.

7.4.1 Vídeos Relacionados

- Procedimentos para Criação de VMs usando os virtualizadores Vmware Player e Virtualbox [6]
- CentOS 8: instalação em VM [7]
- Debian 11: instalação em VM [8]

[6] https://youtu.be/TnCmMTok7Go?si=mmy6Smrx2kOx65BK
[7] https://www.youtube.com/watch?v=_mYl4qOIWWo
[8] https://www.youtube.com/watch?v=iPsIP1HjUlQ

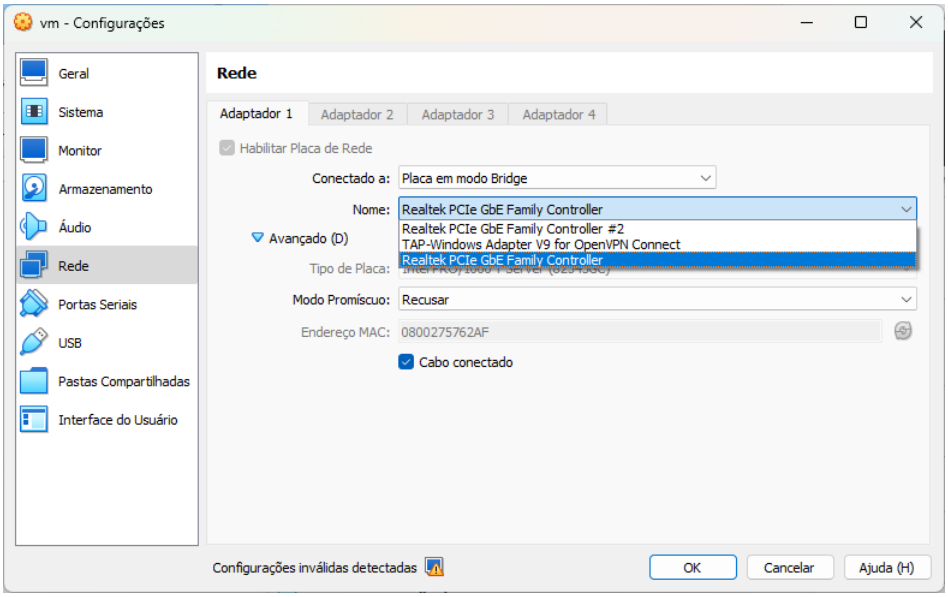

Figure 7.5: VirtualBox: opções de escolha de interface de rede do hospedeiro para ser usada em modo bridged.

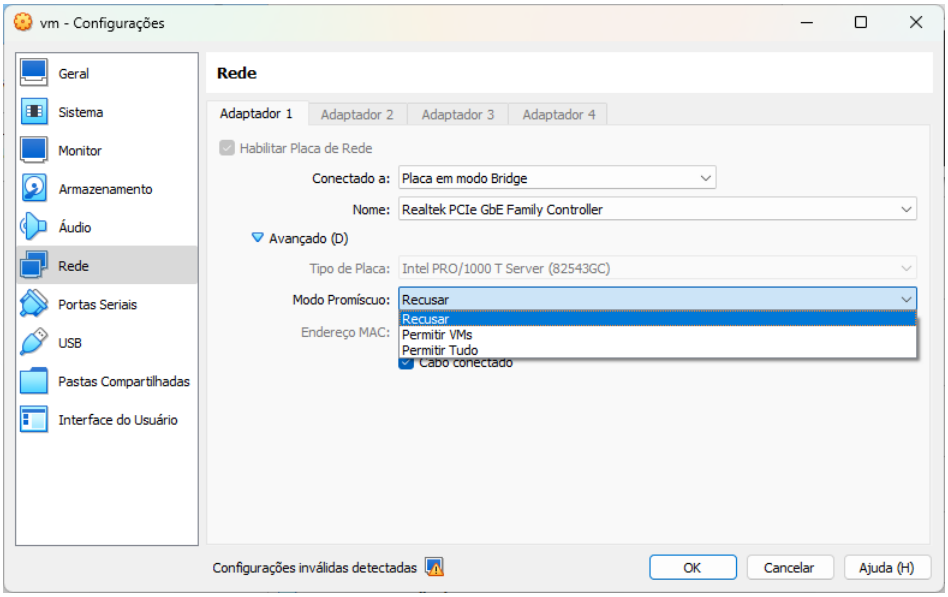

Figure 7.6: VirtualBox: opções de uso da interface bridged em modo promíscuo.

- NDT - Testador de Largura de Banda em Lan - Dicas para Importar a VM [9]

7.5 Tabelas auxiliares

Table 7.1: Servidores de DNS Públicos

Provedor	DNS Primário	DNS Secundário
Google	8.8.8.8	8.8.4.4
Control D	76.76.2.0	76.76.10.0
Quad9	9.9.9.9	149.112.112.112
OpenDNS Home	208.67.222.222	208.67.220.220
Cloudflare	1.1.1.1	1.0.0.1
CleanBrowsing	185.228.168.9	185.228.169.9
Alternate DNS	76.76.19.19	76.223.122.150
AdGuard DNS	94.140.14.14	94.140.15.15

Fonte: Public DNS Server List - https://public-dns.info/ [10]

Table 7.2: Endereços IPv4 para uso privado

Prefixo	Início	Final
10.0.0.0/8	10.0.0.1	10.255.255.255
100.64.0.0/10	10.64.0.1	10.64.127.255
172.16.0.0/12	172.16.0.1	172.16.31.255
192.0.0.0/24	192.0.0.1	192.0.0.255
192.0.2.0/24	192.0.2.1	192.0.2.255
192.31.196.0/24	192.31.196.1	192.31.196.255
192.52.193.0/24	192.52.193.1	192.52.193.255
192.88.99.0/24	192.88.99.1	192.88.99.255
192.168.0.0/16	192.168.0.1	192.168.255.255
192.175.48.0/24	192.175.48.1	192.175.48.255
198.18.0.0/15	198.18.0.1	198.19.255.255
198.51.100.0/24	198.51.100.1	198.51.100.255
203.0.113.0/24	203.0.113.1	203.0.113.255

Fonte: IANA IPv4 Special-Purpose Address Registry -
https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml

[9] https://www.youtube.com/watch?v=k3RKwYuAGik
[10] https://public-dns.info/

[11]

7.6 Características de Rede

[12]

- **Largura de Banda (Bandwidth)**
  - Medida em bit/s
  - Quanto mais, melhor (limite financeiro e tecnológico)
- **Vazão (Throughput)** Quantidade de dados transmitidos por segundo, sem contar com os bytes usados pelos cabeçalhos
- **Goodput** Quantidade de dados transmitidos por segundo, sem contar com os bytes usados pelos cabeçalhos e retransmissões
- **Latência (atraso, delay, latency)**
  - Medido em ms (1/1000 segundos)
  - Inserção de atrasos em todo o caminho do pacote
- **Jitter (variação do atraso)**
  - Medido em ms (1/1000 segundos)
  - Variação do atraso
  - Normalmente, devido ao congestionamento e tempo nas filas dos roteadores
- **Perdas de pacotes (Packet loss)**
  - Medido em % do total de pacotes
  - Perdas podem ser causadas por congestionamento, meios físicos defeituosos, dispositivos com problemas, interferências

7.7 Diagnóstico de Problemas em Rede

Existem diversas metodologias para diagnóstico. O principal objetivo é diminuir o tempo de resolução do problema e aumentar o tempo de disponibilidade da rede, registrando os procedimentos de forma organizada para criar uma base de conhecimento.

As ferramentas de rede são fundamentais para auxiliar no diagnóstico e na validação da resolução do problema. O descarte de hipóteses da origem do problema passa por uma abordagem onde as verificações mais simples acontecem primeiro. Desta forma, haverá uma aceleração para encontrar o problema e começar a etapa de investigação do motivo que originou a falha ou parada em algum serviço de rede.

Na Figura 7.7 um exemplo de metodologia baseada no livro “Diagnosticando Redes - Cisco Internetwork Troubleshooting” de Laura Chappell e Dan Farkas para diagnóstico de problemas em rede. A origem do problema poderá ser alertada por meio de ferramentas de gerenciamento de rede, informada por meio de sistema de tickets ou comunicada por telefone por usuários. Raramente haverá uma informação completa vindo dos usuários. Por exemplo, um relato tal como "Nada funciona!" ou "Não consigo acessar o site XYZ" ajudará o suporte em TI na definição do problema. Portanto, a primeira etapa é "Definir o problema", se é algo localizado ou geral, por exemplo.

[11] https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml
[12] https://www.youtube.com/watch?v=1NmaFjxuqGk

A próxima etapa "Coletar os fatos" passa por analisar os sensores e monitoramentos disponíveis (logs, dashboards, consoles de equipamentos), entrevistar os usuários que relataram o problema, validar se houve alguma atualização de versão em sistemas operacionais, firmwares de dispositivos ou ativação de algum novo serviço. A partir disto, na etapa de "Considerar as possibilidades" haverá o descarte, por meio de validações, do que não está causando o problema. Nesta etapa deve-se considerar o uso das ferramentas corretas para auxílio no diagnóstico. Por exemplo, utilizar a ferramenta ping para validar se o acesso a um banco de dados está disponível não é um teste completo. Para um caso como este, deve-se utilizar um cliente do banco de dados para validar a conexão, autenticação e permissões de acesso do usuário ao banco de dados. Na etapa de "Criar um plano de ação" o responsável pelo suporte ao problema deverá fazer um checklist dos procedimentos a serem realizados e analisar as alternativas para cada resultado dos procedimentos. Após isto, em "Implementar um plano de ação" serão executados os procedimentos planejados e observados os resultados (etapa "Observar os resultados").

Se em algum dos procedimentos executados houver a resolução do problema, deve-se validar com o usuário e com ferramentas que possibilitem coletar evidências do funcionamento correto (tcpdump, logs das aplicações, clientes de acesso ao serviço, acesso à Internet). Encerrada a validação, considera-se o problema resolvido e antes do encerramento haverá a etapa "Documentar os fatos", a etapa mais importante que serve como registro na base de conhecimento do suporte para que em outros casos similares a resolução seja mais rápida e assertiva.

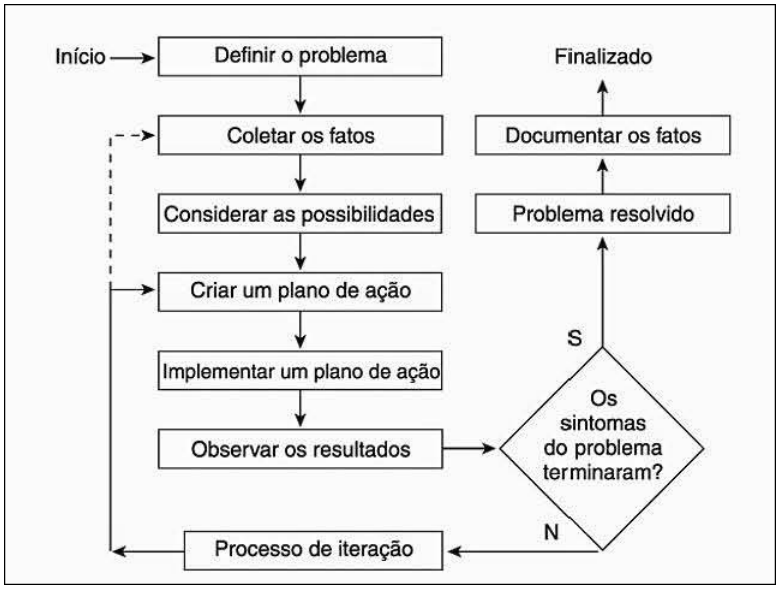

Figure 7.7: Fluxograma para diagnóstico e resolução de problemas em rede.

Outra abordagem para o diagnóstico de problemas em rede está nestas 8 etapas que são complementares a metodologia da Figura 7.7:

1. **Coleta de Informações**
   - Identificar a natureza do problema: constante ou intermitente
   - Verificar quais dispositivos estão afetados e quando o problema começou
   - Anotar alterações recentes na rede ou no sistema
2. **Verificação Física**
   - Confirmar se os dispositivos estão conectados e ligados corretamente
   - Checar os cabos para danos ou desconexões
   - Observar os LEDs dos equipamentos de rede (indicadores de status)
3. **Teste de Conectividade Básica**

- Utilizar comandos básicos para diagnosticar a conectividade:
  - `ping` – Verifica a acessibilidade de um dispositivo
  - `tracert/traceroute` – Identifica possíveis falhas ao longo do caminho de conexão
  - `nslookup/dig` – Testa a resolução de nomes DNS

4. **Isolamento do Problema**
   - Determinar o escopo do problema:
     - Dispositivo específico – Testar em outro equipamento
     - Rede local (LAN) – Verificar comunicação com outros dispositivos na mesma rede
     - Internet – Confirmar acesso a sites externos
5. **Verificação de Configurações**
   - Inspecionar configurações de rede, como:
     - Endereço IP (checar conflitos ou configurações incorretas)
     - Gateway padrão e DNS
     - Políticas de segurança e regras de firewall
6. **Análise de Equipamentos de Rede**
   - Acessar roteadores ou switches para verificar logs de erros
   - Monitorar tráfego de rede para identificar sobrecargas ou anomalias
   - Reiniciar dispositivos, se necessário
7. **Testes e Soluções**
   - Aplicar correções e validar:
     - Atualizar firmware de dispositivos de rede
     - Trocar cabos ou ajustar canais Wi-Fi para reduzir interferências
     - Reconfigurar endereçamento IP ou DNS
8. **Escalamento e Registro**
   - Caso o problema persista:
     - Escalar para suporte técnico especializado
     - Documentar todas as etapas realizadas, incluindo resultados e logs de erros